SEIV Modding 101

Written by: Imperator Fyron (Nolan Kelly), with contributions from Suicide Junkie (Nick Dumas) and Ed Kolis.

Valid as of: Space Empires IV Gold Patch 4+ (v1.91+)

Last updated on April 15, 2006 at 9:11 AM PST.

For updates, please visit the offical website: http://se4modding.spaceempires.net/

Welcome to the Space Empires IV modding tutorial! To go to a specific chapter, click on it in the Table of Contents. You can click on the Chapter Headers to return to the Table of Contents. If you have further questions that are not answered by this document, you can ask them on SpaceEmpires.net ModWorks (http://modworks.spaceempires.net/) and receive expert answers. Comments and questions about the tutorial are welcome as well.

Table of Contents

Chapter 1: Getting Started

The data used by SEIV is stored in convenient text files. The most important of these are located in the Data subfolder of the SEIV folder. These files are:

- Abilities.txt
- CompEnhancement.txt
- Components.txt
- Cultures.txt
- DefaultColonyTypes.txt
- DefaultDesignTypes.txt

- DefaultStrategies.txt
- Demeanors.txt
- EmperorNames.txt
- EmperorTitles.txt
- EmpireNames.txt
- EmpireTypes.txt
- Events.txt
- Facility.txt
- Formations.txt
- Happiness.txt
- IntelProjects.txt
- PlanetSize.txt
- QuadrantTypes.txt
- RacialTraits.txt
- RepairPriorities.txt
- SectType.txt
- Settings.txt
- StellarAbilityTypes.txt
- SystemNames.txt
- SystemTypes.txt
- TechArea.txt
- VehicleSize.txt

These files will be explained in greater detail later on. All that you need to make modifications to SEIV is a text editor. Wordpad works nicely. I suggest that you do not use Notepad, because it is a poorly written program that eats up system resources. If you use it too much, it can lock itself up, forcing you to reboot your computer to be able to continue working.

The way that SEIV loads mods is quite simple. In the folder where the SEIV executable is located (the folder where you installed SEIV), there is a text file called Path.txt. This file has one working line:

```
Using Mod Directory                                                    :=
None
```

Mods are located in subfolders of the SEIV folder. To load a mod, simply change "None" in this line to the name of the folder in which the mod is located. For example, if you have the TDM Modpack installed, you would replace "None" with "TDM-Modpack." The line would then look like this:

```
Using Mod Directory                                                    :=
TDM-Modpack
```

Now, save Path.txt and then load up SEIV. It will load the TDM mod instead of the default SEIV files.

If you want to make a mod, you will need to create a subfolder under the Space Empires IV folder that is named appropriately, such as Proportions or Pirates&Nomads. When using mods, SEIV first looks in the mod's folder. If the game does not find the files that it needs in there, it will use the ones from the default folders. The only exception is the Data folder. Every file from the Data folder must be present in the mod's Data folder, or SEIV will choke. When creating a mod,

you will first need to copy the Data folder from the default folder into the new mod folder. Then, you can begin making alterations. You should never modify the default Data files (except certain lines in Settings.txt that only make cosmetic changes and do not alter game play). Always make your changes in a mod folder.

In most data files, there must be one blank line between all components, facilities, etc. If more lines are placed, the game will stop reading the file and not process any more info from it. Also, there must be one blank line before the first entry. Look in Components.txt to see what I mean. Keep the formatting the same as the default files. Some files, like Settings.txt, do not have extra blank lines. Do not add them.

Pictures for everything are located in the Space Empires IV\Pictures\XXX folders. When a mod is loaded, SEIV will look in the Pictures folder within the mod folder first. If it does not find the required pictures in there, it will look in the default pictures folders. So, it is unnecessary to copy any of the default pictures into your mod. For a good collection of custom pictures, I recommend that you check out the Image Mod, located at http://www.geocities.com/hohoho611ca/imagepack.html.

SE4 checks each turn to make sure that the data files currently used are the same ones as the host used to generate the turn. In a single player sequential movement game, this checking is not really enforced. But in a simultaneous movement game (such as most multiplayer games), data file mismatches can cause you to be unable to load the save game. This is done to prevent cheating. Some data files are checked, while others are not. The following table lists which files are checked and which are not.

File	Incompatibility?	Other Notes
Abilities.txt	No	This file is never used by SE4. It is a reference file for modders only.
CompEnhancement.txt	Yes	n/a
Components.txt	Yes	n/a
Cultures.txt	Yes	n/a
DefaultColonyTypes.txt	No	Only used when a players create their empire files.
DefaultDesignTypes.txt	No	Only used when a players create their empire files.
DefaultStrategies.txt	No	Only used when a players create their empire files.
Demeanors.txt	No	Only used when a players create their empire files.
EmperorNames.txt	No	Only used when a players create their empire files.
EmperorTitles.txt	No	Only used when a players create their empire files.

EmpireNames.txt	No	Only used when a players create their empire files.
EmpireTypes.txt	No	Only used when a players create their empire files.
Events.txt	Yes	n/a
Facility.txt	Yes	n/a
Formations.txt	Yes	n/a
Happiness.txt	Yes	n/a
IntelProjects.txt	Yes	n/a
PlanetSize.txt	Yes	n/a
QuadrantTypes.txt	No	Only used when maps are generated.
RacialTraits.txt	Yes	n/a
RepairPriorities.txt	No	Only used when a players create their empire files.
SectType.txt	Yes	n/a
Settings.txt	Maybe	Some settings cause incompatibility, some do not.
StellarAbilityTypes.txt	No	Only used when maps are generated.
SystemNames.txt	No	Only used when maps are generated.
SystemTypes.txt	No	Only used when maps are generated.
TechArea.txt	Yes	n/a
VehicleSize.txt	Yes	n/a

Return to Table of Contents

Chapter 2: Settings.txt

First, we shall look at the all-important Settings.txt. In this file are the basic parameters that you can change that can have profound effects upon the game.

Most of the lines in Settings.txt are self-explanatory. Take the first one for example.

```
Allow CD Music
:= TRUE
```

This can have one of two values: TRUE or FALSE. If TRUE, then the game will play the music on the SEIV CD. If FALSE, it will not. Most of the lines are as simple as this. If set to TRUE, it is possible that SEIV will crash on certain systems if using either an Audio CD or the SEIV CD. If this happens, SEIV will stop responding after about 30% loaded.

However, some of them are not. We shall now take a look at them.

```
Maximum Number Of Systems
:= 100
```

Although it is obvious what this line does, it is very, very important. When making a new game, the 3 map sizes, Small, Medium and Large, are proportionally based upon this number. A Large quadrant will have about 100% of this number. A Medium quadrant will have 66% of this number. A Small quadrant will have 33% of this number. The highest allowable value for this line is 255. Any number larger than this will cause the game to crash.

```
Minimum Empire Minerals Generation
:= 200
```

This line tells the game that if an Empire would otherwise make no resources (i.e.: it has no Mineral Miners) then it will instead make 200 per turn.

```
Scrap Facility Percent Returned
:= 30
```

This line is a percentage value. In the example above, when you scrap a facility, you will be refunded 30% of the facility's cost.

```
Planet Value Low Percent
:= 0

Planet Value High Percent
:= 150
```

These lines determine the possible range of values that randomly generated planets can have. In the example above, the 3 resource values can range from 0 to 150.

```
Planet Value Low Resources
:= 0

Planet Value High Resources
:= 500000
```

These lines are similar to the last ones, except that these determine the range of values for randomly generated planets in Limited Resource games.

```
Plr Planet Value Low Percent
:= 80

Plr Planet Value Medium Percent
```

```
:= 100

Plr Planet Value High Percent
:= 120
```

These lines determine the average values of Homeworlds. They can vary by as much as 5% from these values.

```
Upgrade Facility Cost Percent
:= 50
```

When you upgrade a facility, the cost is determined by this line. This value is the percentage of how much it would cost just to build the facility.

```
Event Percent Chance Low
:= 5

Event Percent Chance Medium
:= 10

Event Percent Chance High
:= 25
```

These are related to how commonly events occur, depending upon the option selected during game setup.

```
Maintenance Cost Amt Per Dead
:= 20000
```

This line does absolutely nothing. It is a legacy from a very early alpha version of SE4.

```
Empire Starting Percent Maint Cost
:= 25
```

This determines the base percentage of a ship's cost paid every turn. The Maintenance Aptitude racial ability is subtracted from this number.

```
Empire Starting Percent Reproduction
:= 10
```

This setting determines the base reproduction rate. The Reproduction racial ability is added to this value directly, as are the modifiers for the conditions and happiness level of a planet.

```
Combat Fighter Group Amount
:= 20

Combat Mine Group Amount
:= 20

Combat Satellite Group Amount
```

```
:= 20
```

These settings determine the size of unit groups that are launched during combat. They are overridden by the empire's strategies.

```
Defending Units Per Population
:= 20

Population Defender Attack Strength
:= 10

Population Defender Hit Points
:= 30
```

These lines determine the number and strength of the Militia that defend planets from invasion. The first line translates into this: population / 20 = # of militia. The other two determine the combat strength of each militia unit.

```
Automatic Colonization Population
:= 0
```

When a colony ship colonizes a planet, this value is added to the population in the cargo bays of the ship to the new colony.

```
Planet Value Percent Loss After Owner Death
:= 10
```

When you glass a planet (destroy the colony on it, without conquering it), it's value decreases by this amount.

```
Retrofit Cost Percent For Comps
:= 120

Retrofit Cost Percent For Comp Removal
:= 30
```

These lines determine the cost of adding and removing components to a ship when using the Retrofit order.

```
Retrofit Max Percent Difference in Cost
:= 50
```

This line is the limit for the difference in total cost between a ship and a new design that you want to retrofit it to. If a ship costs 10000 resources, then any design that it is legal to retrofit it to can cost a maximum of 15000 resources.

```
Empire Base Planet Mineral Usage Rate
:= 2000

Empire Base Planet Organic Usage Rate
```

```
:= 2000

Empire Base Planet Radioactive Usage Rate
:= 2000
```

These lines determine the base build rate of planets without a space yard.

```
Construction Queue Emergency Build Rate Percent
:= 150

Construction Queue Slow Build Rate Percent
:= 25
```

These are the lines that determine how much the build rate is modified by when a space yard is giving the Emergency Build order.

```
Damage Points To Kill One Population
:= 10
```

This line determines how much damage it takes to kill 1 million population.

```
Population Required to Operate One Facility
:= 50
```

This setting is not implemented in the game code. It does absolutely nothing.

```
Number Of Population Modifiers
:= 12

Pop Modifier 1 Population Amount
:= 99

Pop Modifier 1 Production Modifier Percent
:= 100

Pop Modifier 1 SY Rate Modifier Percent
:= 100
```

These lines are what determines the population modifiers on resource production and space yard rate. The first line must equal the number of modifiers in the file. There can be more or less than 12. The settings are counter-intuitive. The Population Amount is the largest population that will receive the Modifier Percent levels, rather than the minimum amount that will get the bonus. A larger population receives the next bonus listed.

```
Characteristic Physical Strength Max Pct
:= 150

Characteristic Physical Strength Min Pct
:= 50

Characteristic Physical Strength Pct Cost
:= 25
```

```
Characteristic Physical Strength Threshold
:= 20

Characteristic Physical Strength Threshhold Pct Cost Pos
:= 100

Characteristic Physical Strength Threshhold Pct Cost Neg
:= 10
```

These lines are what determines the cost of purchasing the racial characteristics when creating an empire. The Pct Cost is how much it costs to raise the trait by 1%, or how many points are given by reducing it by 1%. The Threshold value means that if you raise or lower the trait by more than (in this case) 20%, then the Threshold Pct Cost kicks in. From that point on, it costs the Threshhold Pct Cost Pos to raise the ability by 1%, and you get Threshhold Pct Cost Neg for each 1% below the Threshold. In this example, Strength can be raised to 120 for 25 points per 1%. For the 121% and above, it costs 100 points per 1%. The trait can be lowered from 100 to 80 for a return of 25 points per 1%. Lowering it to 79% only yields 10 points.

```
Mood Riot Modifier
:= 0

Mood Angry Modifier
:= 80

Mood Unhappy Modifier
:= 90

Mood Indifferent Modifier
:= 100

Mood Happy Modifier
:= 110

Mood Jubilant Modifier
:= 120
```

These lines determine what a colony's happiness level does to it's production. The base production amount is multiplied by these percents. In this example, an Indifferent colony produces at normal capacity. A Jubilant colony produces 20% more than an Indifferent one.

```
Number of Quick Start Styles
:= 16

Quick Start Style 1
:= Amonkrie
```

These lines determine how many races are possible in a Quick Start game, and which races these are.

```
Num Intro Songs
:= 1

Intro Song 1 Track
:= 9

Num Background Songs
```

```
:= 8
Background Song 1 Track
:= 2
Num Combat Songs
:= 6
Combat Song 1 Track
:= 4
```

With these lines, you can use a different music CD while playing SEIV to have different songs play in the background. You can add more tracks if there are more than 8 on the CD that you want to use.

```
Combat Base To Hit Value
:= 100
```

This is the chance to hit an enemy ship at range 0. Increasing it makes it easier in general to hit ships. Decreasing it makes it harder to hit ships.

```
Combat To Hit Modifier Per Square Distance
:= 10
```

This value is subtracted from the Base To Hit Value for each square that is between the firing ship and it's target. If the target is 4 squares away, then the ship receives a -40 penalty to hit.

```
Intelligence Defense Modifier Percent
:= 120
```

For purposes of resolving Counter Intelligence Projects, the amount of points that the defender has put into their project(s) is modified by this percentage value. For example, if an empire has put 10000 points into a Counter Intelligence Project, they would count as 12000 points for purposes of seeing if enemy intelligence projects can overcome it.

```
Ground Combat Damage Modifier Percent
:= 30
```

This modifier is used to increase or decrease the amount of damage done in ground combat (to both sides). Its a good way to speed up or slow down ground combat.

```
Number Of Space Combat Turns
:= 30
Number Of Ground Combat Turns
:= 10
```

These lines determine the maximum number of turns that can occur in one combat. Setting Number Of Ground Combat Turns to 1 or 2 makes it so that reinforcements on both sides can be dropped more easily on the besieged planet. Also, if the combat takes more than one Game Turn, the planet can build Troops to fight in its defense.

```
Ram Ship Source Modifier Percent
:= 60

Ram Ship Target Modifier Percent
:= 100
```

These values modify the damage done to the opposing ship when a Ram order is executed. The Ram Ship Source is applied to the damage done by the ship that is doing the ramming (the one that executed the Ram order). The Ram Ship Target is applied to the ship that is getting rammed (the target of the Ram order). The damage done by each ship in the ram is equal to the total remaining structure of the ship (its hit points), with shields not included in this figure. Damaged ships do less damage in a ram. This total is then modified by the appropriate value, Source for the rammer or Target for the rammee. Damage done by the ramming ship is Normal damage. Damage done by the rammed ship (target) is Skips Shielding damage. So, shields are ignored for the damage done to the ramming ship, but the damage done to the rammed ship is applied to shielding first. If the ramming ship has Warheads on it, then the Warhead damage is added to total damage that it does.

With the default values, say you have a ramming ship with 400 structural hit points and 400 shield points, and a target ship with 300 structural hit points and 400 shield points. The ramming ship will do 240 damage to the target ship. The target ship will do 300 damage to the ramming ship. After the ram, the ramming ship will have 100 structural hit points and 400 shield points left (assuming no shield generating components were destroyed) and the rammed ship will have 300 structural hit points and 160 shield points left.

```
AI Uses Mega Evil Empire
:= True

AI Mega Evil Empire Threshold Score Thousands
:= 500

AI Human Mega Evil Empire Score Percent
:= 170

AI Computer Mega Evil Empire Score Percent
:= 250
```

These lines determine when (and if) an empire triggers the Mega Evil state. While in this state, all AI empires will declare war upon the Mega Evil empire, and will not accept peace. The empire must have a score of at least the "Threshold Score Thousands" value. It's score must be at least "Human Mega Evil Empire Score Percent" as much as the 2nd place empire.

```
Home System Percentage Value With No Spaceport
:= 25
```

This is used only in score calculations. If the Home System of an empire loses it's spaceport, then it still adds 25% of it's score value to the empire's score.

```
Random Player Personality Groups
:= 4

Random Player Personality Group 1 Percent
:= 30
```

```
Random Player Personality Group 2 Percent
:= 30
Random Player Personality Group 3 Percent
:= 30
Random Player Personality Group 4 Percent
:= 10
```

These have to do with AI empires selected during a game setup with random empires.

```
Captured Ship Additional Reload Combat Turns
:= 10
```

If a ship is boarded and captures, it must wait this many combat turns before it can fire again.

```
Maximum Population For Abandon Planet Order
:= 50
```

You cannot give the Abandon Planet order to a planet that has more than this level of population.

```
Population Mass
:= 5
```

This is how many Kilotons of cargo space one unit of population takes up. One unit of population is 1 million people.

```
Reproduction Check Frequency
:= 1
```

Changing this value can slow down the rate of reproduction. At the default setting, this causes the population growth rate per turn to equal the planet's Reproduction Rate / 10. Increasing this value will slow down reproduction. If set at 10, as in Proportions, then the growth rate is only 1 / 10 the normal value. The Reproduction Rate then becomes the population growth per 10 years instead of per 1 year. It functions according to this formula:
pop growth per turn = planetary reproduction rate / 10 / reproduction check frequency

```
Minimum Computer Player Low Setting
:= 1
Maximum Computer Player Low Setting
:= 3
Minimum Computer Player Medium Setting
:= 3
Maximum Computer Player Medium Setting
:= 7
Minimum Computer Player High Setting
:= 6
Maximum Computer Player High Setting
```

```
:= 10

Minimum Neutral Player Low Setting
:= 1

Maximum Neutral Player Low Setting
:= 3

Minimum Neutral Player Medium Setting
:= 3

Maximum Neutral Player Medium Setting
:= 7

Minimum Neutral Player High Setting
:= 6

Maximum Neutral Player High Setting
:= 10
```

These determine the range of the number of computer opponents generated when creating a game with a random number of AI players.

```
Fighter Supply Usage Per Turn
:= 5
```

This setting controls how many supplies a fighter in space (not in cargo of a ship or planet) uses per game turn.

```
Fighters Can Be Hit By Mines
:= True
```

This controls whether fighters can hit mines or not. If set to false, fighters can enter sectors with enemy mines in them and they will not trigger them.

```
Maximum Planet Percent Value
:= 250

Minimum Planet Percent Value
:= 0
```

These settings control the range of values that a planet can ever have in an infinite resources game. The most common way that planet values change is through the use of Value Improvement Plants. With the sample settings, a Value Improvement Plant cannot raise the value of any planet above 250%.

```
Maximum Planet Resource Value
:= 999000000

Minimum Planet Resource Value
:= 0
```

These settings control the range of values that a planet can ever have in a finite resources game. The most common way that planet values change is through the use of Value

Improvement Plants. With the sample settings, a Value Improvement Plant cannot raise the value of any planet above 999000000 points.

```
Bases Can Join Fleets
:= False
```

This setting controls whether bases can or cannot join Fleets. If they can join fleets, they will share their infinite supplies with any ships that join the fleet. The point of bases having infinite supply is so that the players (and AIs) do not have to worry about resupplying an immobile base. The only way to prevent bases from creating infinite supplies for ships (while leaving them with the vehicle type of base and not making them Ships with 0 max engines) is to prevent them from joining fleets in the first place.

```
No Retrofit Adding Of Spaceyards
:= True

No Retrofit Adding Of Colony Module
:= True
```

With these set to true, it is impossible to retrofit a ship from a design with no (Space Yard or Colony Module) to one with a (Space Yard or Colony Module).

```
Seeker Combat Defense Modifier
:= 40
```

This setting controls the inherent penalty to be hit of seekers (such as Capital Ship Missiles). Since this can not be modded for an individual seeker component, this global setting was added.

```
Planet Combat Offense Modifier
:= 30

Planet Combat Defense Modifier
:= -200
```

These control the base bonus to hit and bonus to hit (respectively) for planets in combat.

Return to Table of Contents

Chapter 3: Ships and Other Vehicles

Ships are the main portion of an empire's military. First, we shall take a look at VehicleSize.txt. This file controls all of the vehicle sizes in the game. Vehicle sizes can not be made obsolete. All available vehicle sizes will be displayed. Take the Escort for example:

```
Name                            := Escort
Short Name                      := Escort
Description                     :=
```

```
Code                          := ES
Primary Bitmap Name           := Escort
Alternate Bitmap Name         := Escort
Vehicle Type                  := Ship
Tonnage                       := 150
Cost Minerals                 := 150
Cost Organics                 := 0
Cost Radioactives             := 0
Engines Per Move              := 1
Number of Tech Req            := 1
Tech Area Req 1               := Ship Construction
Tech Level Req 1              := 1
Number of Abilities           := 1
Ability 1 Type                := Combat To Hit Defense Plus
Ability 1 Descr               := Small size makes ship 40% harder to
hit in combat.
Ability 1 Val 1               := 40
Ability 1 Val 2               := 0
Requirement Must Have Bridge  := True
Requirement Can Have Aux Con  := True
Requirement Min Life Support  := 1
Requirement Min Crew Quarters := 1
Requirement Uses Engines      := True
Requirement Max Engines       := 6
Requirement Pct Fighter Bays  := 0
Requirement Pct Colony Mods   := 0
Requirement Pct Cargo         := 0
```

The first line of the file is the name of the ship that is displayed in-game on most screens. The Short Name is displayed in certain areas. The description, which is blank in the standard files, will be displayed when viewing a ship's hull.

```
Code                          := ES
```

This is displayed in several report screens where there is not enough space to place the full name. It is best to not make it longer than 2 characters.

```
Primary Bitmap Name           := Escort
Alternate Bitmap Name         := Escort
```

SEIV looks for graphics in this order:

- Primary Bitmap in the Race's folder
- Secondary Bitmap in the Race's folder
- Primary Bitmap in the Default Race folder
- Secondary Bitmap in the Default Race folder

So you can use custom images for your ship sizes. It is best to leave the Secondary Bitmap Name to a default image, in case a race doesn't have the Primary Bitmap.

```
Vehicle Type                    := Ship
```

This line is very important. It tells the game what type of vehicle it is. The only allowable values here are: Ship, Base, Fighter, Satellite, Mine, Troop, Weapon Platform, Drone.

```
Tonnage                         := 150
Cost Minerals                   := 150
Cost Organics                   := 0
Cost Radioactives               := 0
```

These lines determine how much space is available on the ship and how much the base cost of the ship is.

```
Engines Per Move                := 1
Requirement Uses Engines        := True
Requirement Max Engines         := 6
```

These lines determine whether or not the vehicle in question can have engines (and therefor move). You can change the Engines Per Move to get vastly different propulsion schemes. By increasing the Standard Movement that engines give ships and increasing the Engines Per Move to varying levels for different ships, you can create different systems of propulsion than the normal game has. Check Pirates & Nomads, which was one of the first mods to use the Quasi-Newtonian Propulsion System, where thrust is proportional to a ship's mass.

```
Number of Tech Req              := 1
Tech Area Req 1                 := Ship Construction
Tech Level Req 1                := 1
```

These lines determine the technologies required to make the ship be able to be built. The Tech Area Req must be a technology that is defined in TechArea.txt. The Tech Level Req can be any whole number. If set to a higher number than TechArea.txt defines, You can add more tech requirements by increasing the Number of Tech Req and copying the other two lines, replacing "1" with sequentially numbered Req #. Here is an example:

```
Number of Tech Req              := 3
```

```
Tech Area Req 1                := Ship Construction
Tech Level Req 1               := 5
Tech Area Req 2                := Physics
Tech Level Req 2               := 1
Tech Area Req 3                := Military Science
Tech Level Req 3               := 2
```

Arbitrary tech requirements have been assigned. Make sure that the Tech Req is no greater than the number of defined Tech Requirements. It can be lower. If it is set to 2 in this example, the ship will be available with Ship Construction 5 and Physics 1. It will not require Military Science.

```
Number of Abilities            := 1
Ability 1 Type                 := Combat To Hit Defense Plus
Ability 1 Descr                := Small size makes ship 40% harder to
hit in combat.
Ability 1 Val 1                := 40
Ability 1 Val 2                := 0
```

These lines define the built-in abilities of the vehicle. Make sure that the the Number of Abilities is not greater than the number of defined abilities. You can add more abilities, exactly like adding more Tech Requirements above.

```
Requirement Must Have Bridge   := True
Requirement Can Have Aux Con   := True
Requirement Min Life Support   := 1
Requirement Min Crew Quarters  := 1
```

These determine the control component requirements for vehicles. It is similar for all types of vehicles. For Mines, Drones, Satellites and Weapon Platforms, the "bridge" components is a Computer Core that has the Bridge ability. You can set these requirements to 0 if you want. However, if you do this for a ship type, and one is built without at least a Bridge, 1 Life Support and 1 Crew Quarter, it will receive the penalties of losing these components. So, if you want to have a ship that does not require one of these components, it is a good idea to build the ability into the hull.

```
Requirement Pct Fighter Bays   := 0
Requirement Pct Colony Mods    := 0
Requirement Pct Cargo          := 0
```

These lines determine the requirements for some special components, Fighter Bays, Colony Modules, and Cargo. They are a percent of the ship's total hull size. You can use these to create components that can take up cargo storage without actually being cargo. To do this, you would have to add, for example, this ability to the component that you want to be able to be substituted for cargo bays:

```
Ability 1 Type          := Cargo Storage
Ability 1 Descr         := Counts as a Cargo Bay.
Ability 1 Val 1         := 0
Ability 1 Val 2         := 0
```

You could, for example, place this on a Space Yard component. With 0 storage, the Space Yard won't be able to store anything. However, you could place it onto a Medium Transport and satisfy the transport's cargo requirement.

Return to Table of Contents

Chapter 4: Components

Now that you know a bit about the mechanics of vehicle sizes, lets take a look at what goes into them: components.

```
Name                    := Rock Colony
Description             := Materials needed to start a colony on a rock
planet.
Pic Num                 := 54
Tonnage Space Taken     := 200
Tonnage Structure       := 200
Cost Minerals           := 2000
Cost Organics           := 1000
Cost Radioactives       := 1000
Vehicle Type            := Ship\Base
Supply Amount Used      := 0
Restrictions            := None
General Group           := Colonizing
Family                  := 5001
Roman Numeral           := 0
Custom Group            := 0
Number of Tech Req      := 1
Tech Area Req 1         := Rock Planet Colonization
Tech Level Req 1        := 1
Number of Abilities     := 2
Ability 1 Type          := Colonize Planet - Rock
Ability 1 Descr         := Can colonize a rock based planet.
Ability 1 Val 1         :=
Ability 1 Val 2         :=
```

```
Ability 2 Type          := Cargo Storage
Ability 2 Descr         := Provides 20kT worth of cargo space.
Ability 2 Val 1         := 20
Ability 2 Val 2         := 0
Weapon Type             := None
```

The Name and Description are easy enough: they are what players can use to identify different components.

```
Pic Num                 := 54
```

This value determines which picture is displayed when looking at the component in-game. The component pictures are located in the Pictures\Components subfolder of the SEIV root folder.

```
Tonnage Space Taken     := 200
```

This line determines how much space that the component will take up when placed in a vehicle. It is measured in Kilotons, an arbitrary unit. Don't be too worried about how much everything "weighs" realistically. Does a Small Fighter really weigh 15,000 tons? I didn't think so.

```
Tonnage Structure       := 200
```

This value determines how much damage the component can take before it is destroyed. For units, components are not destroyed individually. Instead, the total structure of all components is added up, and that is how much damage it takes to destroy the unit entirely.

```
Vehicle Type            := Ship\Base
```

This determines what types of vehicles the component can be place on. The allowable values for this field are as follows: Ship, Base, Fighter, Satellite, Mine, Troop, Drone, WeapPlatform, Ship\Base, Ship\Base\Sat, Ship\Base\Sat\Drone, Ship\Base\Sat\WeapPlat\Drone, Ftr\Trp, Ship\Base\Drone, All

```
Vechicle List Type Override := Ship, Base, Fighter, Satellite, Troop,
Drone, WeapPlatform
```

The typo in "Vechicle" is hard-coded into the game executable. This field is optional. If it is present, the field "Vehicle List Type Description" is required. If this field is present, the Vehicle Type field will be ignored. The component will be useable on any vehicle type in the comma separated list. In this example, the component can be placed on anything except for a mine.

```
Vehicle List Type Description := All But Mine
```

This field is optional. If it is present, the field "Vechicle List Type Override" is required. If this field is present, the Vehicle Type field will be ignored. When displaying a component in-game, the text of this field will be displayed in the "Vehicle Types" field of a component description.

```
Supply Amount Used     := 0
```

This value has several different possible functions, depending on the component type. For weapons and components that can be "used" (such as Stellar Manipulation components), it is how many supplies are removed from the vehicle's supply stores. This only pertains to Ships, Fighters and Drones. Other vehicle types do not use supplies. For some component types, like Cloaking Devices, this is how many supplies are used per turn that it is "activated". For engines, only components that have the Standard Ship Movement ability contribute to how many supplies are used when the ship moves. Basically, sum up the Supply Amount Used value for every component on the ship that provides at least one Standard Ship Movement point (not bonus move, or extra movement generation, or combat movement). This value is how many supplies are used each time the ship moves one sector on the system map. As multiple sectors can be traversed in a turn, the supplies used in a turn of movement will be the sum times sectors moved.

```
Restrictions           := None
```

The restrictions are used to set the maximum number of components of a given family that can be placed on a ship. The AI will ignore this setting, however, so care must be taken with their ship designs so that they will not go over the limits. The values that work are None, One Per Vehicle, Two Per Vehicle, Three Per Vehicle, Four Per Vehicle Five Per Vehicle, Six Per Vehicle, Seven Per Vehicle, Eight Per Vehicle, Nine Per Vehicle, and Ten Per Vehicle.

```
General Group          := Colonizing
```

This is what determines where the component goes in the Design Window when you sort the components by type.

```
Family                 := 5001
```

This line tells SEIV what components belong in the same family. Two components are in the same family if they have the same family number. It is also used to determine what gets hidden when the user selects the "Only Latest" button. When this button is selected, only the last available component in the family is displayed in the construction window, as long as there is nothing in between them. If there is another component between two components of the same family, then both will be displayed when "Only Latest" is selected.

```
Roman Numeral          := 1
```

This is an aesthetic label that tells the user which components within a family are "more advanced" than others. The higher the Roman Numeral, the more advanced it is. Unlike with facilities, the Roman Numeral for components has no in-game effect other than what numeral is displayed over the mini component image. Numbers up to 30 will be displayed as roman numerals (e.g.: XXX). Numbers above 30 will be displayed as Arabic Numerals (e.g.: 31). The Name of a component should end with the Roman Numeral assigned to it, unless it is 0. If the Roman Numeral is 0, then there is no Roman Numeral assigned to it. This is used for components that are the only member of their family, like Colony Modules.

```
Number of Tech Req   := 1
Tech Area Req 1      := Rock Colonization
Tech Level Req 1     := 1
```

These lines determine the technologies required to be able to build the facility. Do not set the Number of Tech Req to a number greater than the number of Tech Reqs defined for the facility.

```
Custom Group           := 0
```

This value is ignored on all components except for one type: Ringworld and Sphereworld components. It is used by the Generators to determine which type of component can be used as a requirement to build the Ringworld or Sphereworld. It is used on the Planetary Gravity Plating and the Hyper-Density Cables to tell the game that those are the components that can be used to build the Ringworld or Sphereworld. This process will be explained in greater detail later on.

=0=

Now, we shall take a look at a weapon. We shall look at the Depleted Uranium Cannon V for an example.

```
Name                   := Depleted Uranium Cannon V
Description            := Medium range cannon which fires large
depleted uranium shells.
Pic Num                := 98
Tonnage Space Taken    := 30
Tonnage Structure      := 30
Cost Minerals          := 300
Cost Organics          := 0
Cost Radioactives      := 25
Vehicle Type           := Ship\Base\Sat\WeapPlat\Drone
Supply Amount Used     := 2
Restrictions           := None
General Group          := Weapons
Family                 := 2027
Roman Numeral          := 5
Custom Group           := 0
Number of Tech Req     := 1
Tech Area Req 1        := Projectile Weapons
Tech Level Req 1       := 5
Number of Abilities    := 0
Weapon Type            := Direct Fire
```

```
Weapon Target          := Ships\Planets\Ftr\Sat\Drone
Weapon List Target Override := Ships, Planets, Fighter, Satellite,
Drone
Weapon List Target Description := Ships\Planets\Ftr\Sat\Drone
Weapon Damage At Rng   := 40 40 40 40 40 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Weapon Damage Type     := Normal
Weapon Reload Rate     := 1
Weapon Display Type    := Torp
Weapon Display         := 19
Weapon Modifier        := 0
Weapon Sound           := uranc.wav
Weapon Family          := 25
```

The weapon has all the same lines except for the Weapon X lines are added to it.

```
Weapon Type            := Direct Fire
```

This tells the game what type of weapon the component is. If set to None, then the component has no weapon properties, and cannot be fired as a weapon. You can design "combo" components, that are both a normal component and a weapon. For example, you could take a Shield Generator and mod in Weapon abilities. This component would both generate shielding and be usable as a weapon. These are the allowable Weapon Types:

```
None, Direct Fire, Seeking, Point-Defense, Warhead
```

None cause the component to not be a weapon.

Direct Fire makes it into a beam type weapon, such as a Depleted Uranium Cannon or an Anti-Proton Beam.

Seeking makes the weapon into a missile weapon. It will fire missiles at targets. An example is the Capital Ship Missile.

Point-Defense makes the weapon into a Direct Fire weapon that fires automatically at enemy targets as soon as they move into range.

A Warhead weapon adds it's damage value to the total damage that the vehicle does to it's target when given the Ram order. Warheads do NOT add damage to a vehicle that gets rammed, only to the attacking vehicle.

```
Weapon Target          := Ships\Planets\Ftr\Sat\Drone
```

This determines what a weapon can target.

```
Weapon List Target Override := Ships, Planets, Fighter, Satellite,
```

```
Drone
```

This field is optional. If it is present, the Weapon Target field will be ignored, and the "Weapon List Target Description" field is required. This determines what a weapon can target using comma separated lists.

```
Weapon List Target Description := Ships\Planets\Ftr\Sat\Drone
```

This field is optional. If it is present, the Weapon Target field will be ignored, and the "Weapon List Target Override" field is required. This field is the text that will be displayed for the Weapon Target field of a component description in-game.

```
Weapon Damage At Rng  := 40 40 40 40 40 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

This line determines what the range of the weapon is and how much damage it does at every possible range. There must be 20 values here. For a seeking weapon, this also determines the range of the seeker in combat, before it "runs out of fuel" and disappears. For troop weapons, range is ignored. All weapons are assumed to be fired at range 1 in troop combat.

You CAN add a 21st value to this line. It has some odd effects. If you make a seeker with a range of 21 or greater, it will have unlimited "fuel" during combat, and will not disappear until it hits its target, the target is destroyed, or the combat ends.

```
Weapon Damage Type    := Normal
```

This value determines what type of damage the weapon does. The possible types of damage are:

```
   Normal, Shields Only, Skips Normal Shields, Only Engines, Only
Weapons,
   Plague Level 1, Plague Level 2, Plague Level 3, Plague Level 4,
Plague Level 5,
   Only Planet Population, Only Planet Conditions, Only Resupply
Depots,
   Only Spaceports, Pushes Target, Pulls Target, Random Target
Movement,
   Only Shield Generators, Only Boarding Parties, Only Security
Stations,
   Only Planet Destroyers, Skips Armor, Skips Shields And Armor,
   Quad Damage To Shields, Increase Reload Time, Disrupt Reload Time,
   Crew Conversion, Only Master Computers, Skips All Shields,
   Double Damage To Shields, Half Damage To Shields, Quarter Damage To
Shields
```

They are all pretty self-explanatory. However, Only X Type (such as Only Shield Generators, Only Engines, etc.) damage is ignored if it cannot destroy a component completely in one hit.

Example: if Engines have 20 structure, an Only Engines damage type weapon that does 15 points of damage *will not* do anything to the target ship. 2 shots will not stack. Since each shot cannot destroy a component entirely, all of the damage gets ignored. SE4 does not keep track of partial damage for Only X Types.

```
Weapon Reload Rate     := 1
```

This determines how long it takes to "reload" the weapon. A value of 1 means that the weapon will be able to fire every round during combat. A value of 2 means that the weapon will be able to fire every other round. For troop weapons, reload rate is ignored. All weapons are assumed to have a reload rate of 1 in troop combat.

```
Weapon Display Type    := Torp
```

This value determines what picture type is displayed when the weapon type is fired in combat.

```
Weapon Display         := 19
```

This determines which graphic from the appropriate file in Pictures\Combat is used to display the weapon's firing.

```
Weapon Modifier        := 0
```

For Direct Fire weapons, this is added to the weapon's chance to hit. If negative, it will make it less likely that the weapon will be able to hit it's targets.

```
Weapon Sound           := uranc.wav
```

This determines which sound is played when the weapon is fired in combat.

```
Weapon Family          := 25
```

This value is used in place of the normal Family value for weapon components.

Return to Table of Contents

Chapter 5: Technologies

Technologies are one of the simplest areas to mod. However, they can have the most profound of effects. For reference, "tech" is an abbreviation for "technology."

```
Name                   := Physics
Group                  := Theoretical Science
Description            := The science dealing with the properties and
interactions of matter and energy.
```

```
Maximum Level          := 4
Level Cost             := 50000
Start Level            := 0
Raise Level            := 1
Racial Area            := 0
Unique Area            := 0
Can Be Removed         := True
Number of Tech Req     := 0

Name                   := Shields
Group                  := Applied Science
Description            := Protective energy barrier which surrounds a
starship or base.
Maximum Level          := 10
Level Cost             := 10000
Start Level            := 0
Raise Level            := 0
Racial Area            := 0
Unique Area            := 0
Can Be Removed         := True
Number of Tech Req     := 1
Tech Area Req 1        := Physics
Tech Level Req 1       := 1

Name                   := Energy Stream Weapons
Group                  := Weapon Technology
Description            := Weapons employing a continuous stream of
energy.
Maximum Level          := 12
Level Cost             := 5000
Start Level            := 0
Raise Level            := 2
Racial Area            := 0
Unique Area            := 0
Can Be Removed         := True
Number of Tech Req     := 1
Tech Area Req 1        := Physics
Tech Level Req 1       := 1
```

We shall take a look at 3 different technologies. Physics, a theoretical technology, allows Shields and Energy Stream Weapons to be researched.

```
Name                    := Physics
```

This line is the name of the technology. It is important to note that this line creates what is called a "Technology Area". Generally, you can research multiple levels in a Tech Area. For this example, this entry has created the Physics Technology Area, which has 4 levels in it.

```
Group                   := Theoretical Science
```

This line determines where the tech is located in the Research Window. Any group name can be used. SEIV will just add a new area for the tech group.

```
Maximum Level           := 4
```

This determines the number of levels in the Tech Area. For this example, the entry has set up Physics 1, Physics 2, Physics 3 and Physics 4.

```
Level Cost              := 50000
```

This is the cost to research level 1 in the Tech Area. It determines the cost to research higher levels, based upon what Tech Cost the game was set up to use.

```
Start Level             := 0
```

This determines the starting number of tech levels each empire receives when starting a game with a "Low Tech Start." The 3 colonization tech areas are a special case which the game will set higher based on your racial traits.

SE4 treats colony module technologies differently than other technologies. It ignores the Start and Raise Levels. 1 level is given in the first tech area in the file that gives the appropriate colony module type, based on what planet type the race's homeworld uses. No other levels in these tech areas are given to the empire. If you make Rock Colonization Technology have 2 max levels and set it's Start Level to 2, non-rock races get no tech levels, and rock races get 1 tech level in Rock Colonization Technology. The Start Level is ignored. In High Tech Start games, all of the available colony techs are given.

```
Raise Level             := 1
```

This determines the starting number of tech levels each empire receives when starting a game with a "Medium Tech Start." If the Start Level is higher than this value, the game will use it instead.

```
Racial Area             := 0
```

This determines whether the technology is a Racial Tech or a Normal Tech. You can add new

Racial Tech areas at will. Make sure to add Racial Abilities for them. The number placed in this field is what determines what racial ability is required to research the technology. A race can not have a racial technology for which it does not possess the required Racial Ability. However, you can mod capturable racial technologies. Here is how it works (assume "you" are the empire without the racial trait):

Tech A is a racial tech
Tech B is a normal tech that requires tech A, level 1
Tech C is a normal tech that requires tech B, level 1

Results:

-you CANNOT steal tech A
-you CANNOT research tech A

-you CAN steal tech B
-you CANNOT research tech B

-you CAN steal tech C
-you CAN research tech C, if you have stolen tech B, level 1

```
Unique Area             := 0
```

This tells the game if the tech is a Unique Tech or a Normal Tech. Unique Techs are the technologies given by colonizing a planet that has an "Ancient Ruins Unique" ability on it. A tech can be both Racial and Unique. This means that if you make a racially Organic tech that is also Unique, this is what will happen: A race colonizes the planet. If the race has the Organic trait, it gets the Unique Tech. If the race isn't organic, it gets nothing.

```
Can Be Removed          := True
```

This tells SEIV whether the Tech Area can be removed from play when setting up a game.

```
Number of Tech Req      := 1
Tech Area Req 1         := Physics
Tech Level Req 1        := 1
```

This determines the prerequisite technologies that will allow you to begin research into a Tech Area. Make sure that Number of Tech Req does not exceed the number of defined tech requirements.

Now that we understand the basics of how to make techs, lets see how they all work together. In our running example, a beginning empire on a Low Tech Start cannot research Shields or Energy Stream Weapons. They can research Physics, a Theoretical Tech. Generally, the techs in SEIV are set up so that you first research a Theoretical Tech, which gives nothing on it's own, but allows Applied Techs and/or Weapons Techs to be researched. Once an empire researches Physics 1, it will be able to research Shields and Energy Stream Weapons (plus a number of other technologies). Researching level 2 in the Physics Tech Area gives access to more Applied and Weapons Techs, such as Phased Energy Weapons.

Chapter 6: Facilities

By now, you should have a basic understanding of how the files work. This chapter shall deal with Facilities. The information in it may begin to sound repetitive.

First, we will take a look at how SE4 determines the starting facilities on a homeworld planet. This is unfortunately hard-coded, and so the starting facility lineup can only be altered indirectly. Here is how it works:

The facilities on the homeworld will be added as follows (all must be available with starting technology):
1) The last facility with the Spaceport ability in the file.
2) The last facility with the Space Yard ability in the file.
3) The last facility with the Resupply ability in the file.
4) One facility that has the highest level of the Resource Generation - Organics ability.
5) One facility that has the highest level of the Resource Generation - Radioactives ability.
6) One facility that has the highest level of the Resource Generation - Minerals ability.
7) One facility that has the highest level of the Point Generation - Research ability.
8) Repeat 6 & 7 until the planet is filled.

Position in the file is irrelevant for the resource/research facilities. Technically, the Mineral facilities are added before the Organics and Radioactives ones (at least the first one is). But, they are presented in this order for ease of reading. There is 0 difference for in-game effects (unless you mod the planet used as a homeworld to only have 4 facility slots, of course).

=0=

Now we shall take a look at a sample facility.

```
Name                := Space Yard Facility I
Description         := Large construction facility which allows the
construction of ships in space.
Facility Group      := Space Yard
Facility Family     := 38
Roman Numeral       := 1
Restrictions        := None
Pic Num             := 16
Cost Minerals       := 10000
Cost Organics       := 0
Cost Radioactives   := 0
Number of Tech Req  := 1
Tech Area Req 1     := Space Yards
Tech Level Req 1    := 1
```

```
Number of Abilities := 4
Ability 1 Type      := Space Yard
Ability 1 Descr     := Can construct with 2000 minerals per turn.
Ability 1 Val 1     := 1
Ability 1 Val 2     := 2000
Ability 2 Type      := Space Yard
Ability 2 Descr     := Can construct with 2000 organics per turn.
Ability 2 Val 1     := 2
Ability 2 Val 2     := 2000
Ability 3 Type      := Space Yard
Ability 3 Descr     := Can construct with 2000 radioactives per turn.
Ability 3 Val 1     := 3
Ability 3 Val 2     := 2000
Ability 4 Type      := Component Repair
Ability 4 Descr     := Can repair 5 components per turn.
Ability 4 Val 1     := 5
Ability 4 Val 2     := 0
```

The Space Yard is a good example to work off of.

```
Name                := Space Yard Facility I
```

Like components, the name of facilities should have the appropriate Roman Numeral on the end, unless it is the only facility in it's family.

```
Facility Group      := Space Yard
```

This line tells SEIV where to place the facility in the list in the Construction Window for planets.

```
Facility Family     := 38
```

This tells SEIV what facilities belong in the same family. Two facilities are in the same family if they have the same family number. This is used to determine what facilities are upgraded to what with the Upgrade button, in conjunction with the Roman Numeral. It is also used to determine what gets hidden when the user selects the "Only Latest" button. When this button is selected, only the last available facility in the family is displayed in the construction window.

```
Roman Numeral       := 1
```

This tells SEIV which facilities within a family are "more advanced" than others. The higher the Roman Numeral, the more advanced it is. A higher Roman Numeral facility within the same family will make a facility upgradable if it is placed immediately after it in Facility.txt. The Space

Yard I can be upgraded to the Space Yard II because they are in the same Family and the SY II has a higher Roman Numeral than the SY I. It can range from 0-20. The Name of a facility should end with the Roman Numeral assigned to it, unless it is 0. If the Roman Numeral is 0, then there is no Roman Numeral assigned to it. This is used for facilities that are the only member of their family, like Spaceports.

```
Restrictions          := None
```

This line is not used by SEIV.

```
Cost Minerals         := 10000
Cost Organics         := 0
Cost Radioactives     := 0
```

These lines determine the cost of building the facility. When upgrading a facility to a more advanced version, the build costs are multiplied by the Upgrade Facility Cost Percent line in Settings.txt to determine the cost to upgrade.

```
Number of Tech Req    := 1
Tech Area Req 1       := Space Yards
Tech Level Req 1      := 1
```

These lines determine the technologies required to be able to build the facility. Do not set the Number of Tech Req to a number greater than the number of Tech Reqs defined for the facility.

```
Number of Abilities := 4
Ability 1 Type      := Space Yard
Ability 1 Descr     := Can construct with 2000 minerals per turn.
Ability 1 Val 1     := 1
Ability 1 Val 2     := 2000
```

These lines determine the abilities that the facility has. Do not set the Number of Abilities to a number greater than the number of Abilities defined for the facility.

Return to Table of Contents

Chapter 7: Quadrants

The Quadrants.txt file determines how the systems are arranged on the Galaxy Map, and how many of each type of system is in a particular map.

There can be as many types of quadrants in the file as you want. We shall take a look at part of the Mid-Life quadrant:

```
Name                        := Mid-Life
Description                 := Standard middle age section of the
galaxy.
Min Dist Between Systems    := 1
System Placement            := Random
Max Warp Points per Sys     := 5
Min Angle Between WP        := 60
Number of System Types      := 39
Type 1 Name                 := Standard 1
Type 1 Chance               := 240
Type 2 Name                 := Standard 2
Type 2 Chance               := 237
Type 3 Name                 := Standard 3
Type 3 Chance               := 195
```

The rest of it is just more system types, and so is repetitive.

```
Name                        := Mid-Life
Description                 := Standard middle age section of the
galaxy.
```

The name is what is displayed when selecting a quadrant type when creating a game. The Description is never displayed anywhere.

```
Min Dist Between Systems    := 1
```

This is the minimum number of system squares on the Galaxy Map that must be between each system. At 1, this means that no 2 systems can be in directly adjacent squares.

```
System Placement            := Random
```

This determines how the systems are set up on the Map. The possible values are: Random, Clusters, Spiral, Diffuse, Grid. Random places the systems in a chaotic fashion. Cluster makes the systems be placed in 12 areas on the map. The systems are concentrated in these 12 groups. The groups are organized with 4 groups in 3 rows on the Map. Spiral makes the systems organized into a spiral-type pattern. Diffuse places the systems in a more organized fashion than Random. Grid sets the systems into a rigid gridded pattern.

```
Max Warp Points per Sys     := 5
```

This determines the maximum warp points that can be in a system. This can be overridden to ensure connectivity.

```
Min Angle Between WP        := 60
```

This determines the minimum angle that the warp lines coming out of a system on the Galaxy Map can make. This can be overridden to ensure connectivity.

```
Number of System Types      := 39
```

This determines how many system types can be in the galaxy.

```
Type 1 Name                 := Standard 1
Type 1 Chance               := 240
Type 2 Name                 := Standard 2
Type 2 Chance               := 237
Type 3 Name                 := Standard 3
Type 3 Chance               := 195
```

These lines determines which system types can be present in the quadrant, and what chance they have of appearing. The Chances are in tenths of a percent. They should total up to 1000 (100%) for the quadrant. However, do not worry to much about getting it at exactly 1000. If the total is 1010, 990, or something like that, SEIV will compensate and readjust the values when it generates maps with that quadrant.

Return to Table of Contents

Chapter 8: Systems

SystemTypes.txt is the file that defines all of the Systems called for in QuadrantTypes.txt.

```
Name                        := Standard 1
Description                 := Standard solar system.
System Physical Type        := Normal
Background Bitmap           := Starmap.bmp
Empires Can Start In        := TRUE
Mask Background Objs        := FALSE
Non-Tiled Center Pic        := FALSE
Number of Abilities         := 0
WP Stellar Abil Type        := Normal Warp Point
Number of System Objs       := 12
Obj 1 Physical Type         := Star
Obj 1 Position              := Ring 1
Obj 1 Stellar Abil Type     := Normal Star
```

```
Obj 1 Size                := Any
Obj 1 Age                 := Any
Obj 1 Color               := Any
Obj 1 Luminosity          := Any
Obj 2 Physical Type       := Planet
Obj 2 Position            := Ring 2
Obj 2 Stellar Abil Type   := Normal Planet
Obj 2 Size                := Any
Obj 2 Atmosphere          := Any
Obj 2 Composition         := Any
Obj 3 Physical Type       := Planet
Obj 3 Position            := Ring 3
Obj 3 Stellar Abil Type   := Normal Planet
Obj 3 Size                := Any
Obj 3 Atmosphere          := Any
Obj 3 Composition         := Any
Obj 4 Physical Type       := Planet
Obj 4 Position            := Ring 3
Obj 4 Stellar Abil Type   := Normal Planet
Obj 4 Size                := Any
Obj 4 Atmosphere          := Any
Obj 4 Composition         := Any
Obj 5 Physical Type       := Planet
Obj 5 Position            := Ring 4
Obj 5 Stellar Abil Type   := Normal Planet
Obj 5 Size                := Any
Obj 5 Atmosphere          := Any
Obj 5 Composition         := Any
Obj 6 Physical Type       := Planet
Obj 6 Position            := Ring 4
Obj 6 Stellar Abil Type   := Normal Planet
Obj 6 Size                := Any
Obj 6 Atmosphere          := Any
Obj 6 Composition         := Any
Obj 7 Physical Type       := Planet
Obj 7 Position            := Ring 5
Obj 7 Stellar Abil Type   := Normal Planet
```

```
Obj 7 Size                := Any
Obj 7 Atmosphere          := Any
Obj 7 Composition         := Any
Obj 8 Physical Type       := Planet
Obj 8 Position            := Ring 5
Obj 8 Stellar Abil Type   := Normal Planet
Obj 8 Size                := Any
Obj 8 Atmosphere          := Any
Obj 8 Composition         := Any
Obj 9 Physical Type       := Storm
Obj 9 Position            := Ring 5
Obj 9 Stellar Abil Type   := Normal Storm
Obj 9 Size                := Any
Obj 10 Physical Type      := Planet
Obj 10 Position           := Ring 6
Obj 10 Stellar Abil Type  := Normal Planet
Obj 10 Size               := Any
Obj 10 Atmosphere         := Any
Obj 10 Composition        := Any
Obj 11 Physical Type      := Planet
Obj 11 Position           := Ring 6
Obj 11 Stellar Abil Type  := Normal Planet
Obj 11 Size               := Any
Obj 11 Atmosphere         := Any
Obj 11 Composition        := Any
Obj 12 Physical Type      := Asteroids
Obj 12 Position           := Ring 6
Obj 12 Stellar Abil Type  := Normal Asteroids
Obj 12 Size               := Any
Obj 12 Atmosphere         := Any
Obj 12 Composition        := Any
```

This standard system is one of the three generic Solar Systems.

```
Description               := Standard solar system.
```

This description is shown when a player clicks on the background of the System Map in the game.

```
System Physical Type        := Normal
```

This determines what basic type of system the system is. The possibilities are: Normal, Nebulae, Black Hole. Basically, Storms and Nebulae use the "Nebulae" type, Black Holes use the "Black Hole" type, and everything else uses "Normal." This setting actually has no real effect on the game.

```
Background Bitmap           := Starmap.bmp
```

This is the picture from Pictures\Systems that is placed as the background when the system is viewed on the System Map. There are 3 possible pictures used. If SEIV is run in 800x600 screen resolution, the picture from Pictures\Systems\800X600 is used. If SEIV is run in 1024x768 or greater screen resolution, the picture from Pictures\Systems\1024X768 is used. The pictures that are in Pictures\System are displayed in the Info Box when a player clicks on the background of a system in the System Map in order to view the special info for the system.

```
Empires Can Start In        := TRUE
```

This tells SEIV if empires can have homeworlds in the system or not. Generally, an extra planet is added to a home system for the homeworlds. This setting can be overridden if every system in the galaxy has this set to FALSE.

```
Mask Background Objs        := FALSE
```

This determines whether objects within the system such as planets, warp points, etc. need to be masked when drawn on the background bitmap.

```
Non-Tiled Center Pic        := FALSE
```

This tells SEIV whether the picture displayed should be shown in the center of the Combat Map and it is not tiled or not. If set to TRUE, the image will be displayed in the center of the Combat Map. If FALSE, it will be tiled in the Combat Map. This means that it will be placed in the top left corner of the map and then copied directly next to it as many times as necessary to fill the map. It will be placed beneath it too.

```
Number of Abilities         := 0
```

This determines the number of abilities inherent to the system that affect everything in every sector (square) of the System Map. This should not be greater than the number of abilities defined immediately following this line.

```
WP Stellar Abil Type        := Normal Warp Point
```

This tells SEIV what type of Warp Points to use from StellarAbilityTypes.txt for the Warp Points generated for the system.

```
Number of System Objs       := 12
```

This determines the number of objects located within the system. This should not be greater than the number of objects defined for the system.

Now we shall look at the coding for individual objects within the system. The following lines must be present for every object:

```
Obj X Physical Type        :=
```

This determines what type of object is being defined. The possible Object Types are: Planet, Asteroids, Storm, Sun, Destroyed Star, Comet.

```
Obj X Position             :=
```

This determines where in the system the object will be located. The possible locations are:
Ring 1 : Center Square of System.
Ring 2 - 7 : 1 - 6 squares out from Center of System.
The actual position in the ring is completely random.
Coord X, Y : A coordinate position within the system.
X is horizontal position from 0 to 12,
Y is vertical position from 0 to 12.
There is no randomness to this placement.
Same As X : The same position as Obj X.
There is no randomness to this placement.
Circle Radius X: A true circle in the system at distance X.
The actual position in the ring is completely random.

```
Obj X Stellar Abil Type    :=
```

This is the type that will be used to determine what random ability the object has. They are defined in StellarAbilityTypes.txt.

```
Obj X Size                 :=
```

This determines the size of the object. It can be: Any, Tiny, Small, Medium, Large, Huge.

For storms, this is all you need. But for other objects, you need more lines. Stars need:

```
Obj X Age                  := Any
Obj X Color                := Any
Obj X Luminosity           := Any
```

Age can be: Any, Young, Average, Old, Ancient.
Color can be: Any, Yellow, Red, Purple, Green, Blue, White, Orange. Luminosity can be: Any, Dim, Average, Bright, Super Bright.

Planets require the following lines:

```
Obj X Atmosphere          :=
```

This is the atmosphere of the planet. It can be: Any, None, Methane, Oxygen, Hydrogen, Carbon Dioxide.

```
Obj X Composition         :=
```

This is the planet's Type. It can be: Any, Rock, Ice, Gas Giant.

Asteroids require the following lines:

```
Obj X Atmosphere          := Any
Obj X Composition         := Any
```

These should be left at Any. Asteroids are all None Rock. Since they cannot be anything else, it is easier to leave these setting on Any.

Return to Table of Contents

Chapter 9: System Objects

This chapter will focus on the objects that can be placed in sectors (the squares on the system map): stars, planets, storms, asteroids, warp points. These objects are defined in SectType.txt. They are given abilities in StellarAbilityTypes.txt. They are called for in SystemTypes.txt. We shall begin by looking at SectType.txt. There can not be more than 1000 objects defined in this file.

```
Allowable Values for Physical Type:
  None, Planet, Asteroids, Storm, Sun, Warp Point, Destroyed Star,
Comet
```

The Physical Type is what defines the basic nature of the object. Different physical types require different additional lines that define the particular object (to be out-lined later).

```
Allowable Values for Planet Size:
  Asteroids, Tiny, Small, Medium, Large, Huge
```

Every planet size called for in PlanetSize.txt must have at least 1 entry in SectType.txt, or else it will never appear in the game. The "Planet Size" line in SectType.txt must match the "Name" line in PlanetSize.txt for appropriate planet types. For planets in SystemTypes.txt that are given a size of Any, the size is determined randomly. The relative number of each size of planet used for the Any sized-planets is proportional to how often that particular size of planet appears in SectType.txt. If 20% of all the planet entries in SectType.txt have a size of Tiny, then 20% of the planets with Any size placed on a randomly generated map will have a size of Tiny. This is how all of the Size, Atmosphere, Color, Planet Physical Type, etc. values set to Any in SystemTypes.txt are determined.

```
Allowable Values for Planet Physical Type:
  Rock, Ice, Gas Giant

Allowable Values for Planet Atmosphere:
  None, Methane, Oxygen, Hydrogen, Carbon Dioxide

Allowable Values for all other sizes:
  Tiny, Small, Medium, Large, Huge

Allowable Values for Star Age:
  Young, Average, Old, Ancient

Allowable Values for Star Color:
  Yellow, Red, Purple, Green, Blue, White, Orange

Allowable Values for Star Luminosity:
  Dim, Average, Bright, Super Bright
```

No values other than those lists for any of these fields may be used. Planets can never have a Physical Type of Gas Giant and Atmosphere of None.

Here is an example of a typical Planet:

```
Physical Type        := Planet
Picture Num          := 0
Description          := Huge planet with an abundance of surface
minerals.
Planet Size          := Huge
Planet Physical Type := Gas Giant
Planet Atmosphere    := Carbon Dioxide
```

This planet will be a Huge Gas Giant with a Carbon Dioxide Atmosphere. The Picture Num determines what picture is displayed for such a planet.

Here is an example of a typical Star:

```
Physical Type        := Star
Picture Num          := 225
Description          := An average yellow star.
Star Size            := Medium
Star Age             := Average
```

```
Star Color            := Yellow
Star Luminosity       := Average
```

The Size, Age, Color and Luminosity do not have any in-game effects. They are merely there for aesthetic purposes. If a particular combination of these values does not appear in SectType.txt, it can not appear in the game (except in custom maps made in the Map Editor). So, a Star (or any object, for that matter) with all values set to Any is not truly random.

Here is an example of a typical Storm:

```
Physical Type         := Storm
Picture Num           := 250
Description           := A stellar storm composed of plasma fields,
energy discharges, and electro-static gasses.
Storm Size            := Medium
Combat Tile           := Storm5
```

The Storm Size has no in-game effects. The abilities of Storms are not determined in this file, but in StellarAbilityTypes.txt. The Combat Tile is the picture used on the combat screen. It is placed in the top left corner. Then, a copy is placed immediately to the right of it. As many tiles as are necessary are placed to fill a row. Then, a copy is placed directly below the top left corner. That row is filled, and so on.

Here is an example of a typical Asteroid Belt:

```
Physical Type         := Asteroids
Picture Num           := 275
Description           := Tiny field of space rocks orbiting through
this system.
Planet Size           := Tiny
Planet Physical Type  := Rock
Planet Atmosphere     := None
Combat Tile           := Asteroids1
```

The Size of the Asteroid determines the largest possible sized-planet that can be constructed from the Asteroids. The only possible Physical Type and Atmosphere of an Asteroid Belt is Rock None. Other values will not work. The Combat Tile is the picture used on the combat screen. It is placed in the top left corner. Then, a copy is placed immediately to the right of it. As many tiles as are necessary are placed to fill a row. Then, a copy is placed directly below the top left corner. That row is filled, and so on.

Here is an example of a typical Warp Point:

```
Physical Type         := Warp Point
Picture Num           := 295
```

```
Description           := A smaller than normal warp point.
Warp Point Size       := Small
Warp Point One-Way    := FALSE
Unusual               := TRUE
```

The Warp Point Size does not have any in-game effects. If set to TRUE, the Warp Point One-Way field makes only one Warp Point instead of the normal two reciprocal Warp Points. Ships can travel through one way, but not the other.

=0=

Now, we shall look at PlanetSize.txt. This file defines the sizes of planets. Here is a sample entry:

```
Name                      := Tiny
Physical Type             := Planet
Stellar Size              := Tiny
Max Facilities            := 5
Max Population            := 500
Max Cargo Spaces          := 1000
Max Facilities Domed      := 1
Max Population Domed      := 100
Max Cargo Spaces Domed    := 200
Constructed               := False
Special Ability ID        := 0
```

This is the entry for a Tiny sized planet.

```
Name                      := Tiny
```

This is the name for the planet type that is displayed in-game. It is also the tag that is referenced by "Planet Size" in SectType.txt for determining the size of a planet entry. This planet will be valid for all SectType.txt entries that call for a Tiny sized planet.

```
Physical Type             := Planet
```

This is the type of object that the entry is. It can either be a Planet or an Asteroid. No other values will work in this file.

```
Stellar Size              := Tiny
```

This is the hard-coded "size" of the object. It can only be Tiny, Medium, Small, Large or Huge.

```
Max Facilities              := 5
Max Population              := 500
Max Cargo Spaces            := 1000
```

These lines determine how many facilities, how much population and how much cargo can be placed/built on a planet that has the same atmosphere as the breathable type of all races on the planet (i.e.: a non-domed colony). These entries are never referenced for an Asteroid type object.

```
Max Facilities Domed        := 1
Max Population Domed        := 100
Max Cargo Spaces Domed      := 200
```

These lines determine how many facilities, how much population and how much cargo can be placed/built on a planet that has a different atmosphere of at least one race's breathable type that is on the planet (i.e.: a domed colony). These entries are never referenced for an Asteroid type object.

```
Constructed                 := False
```

This tells SE4 whether the planet is a special type that can only be constructed (e.g.: Ringworlds). Constructed planets will never appear randomly on a map.

```
Special Ability ID          := 0
```

This number is used to determine what type of planet is constructed when a component with the Create Constructed Planet ability is used. A Sphere World Placement Generator has an ability of 2, so it will construct the planet with a Special Ability ID of 2, which is the Sphereworld.

=0=

Now, we shall take a look at StellarAbilityTypes.txt. This file defines the possible abilities that sector objects can have. Here is a sample:

```
Name                        := Normal Planet
```

This defines the type of object (called for in SystemTypes.txt) that is being defined.

```
Number of Poss Abilities    := 7
```

This defines the possible abilities that the object can have. Each randomly generated object can only have 1 ability (though is not required to have an ability). Make sure that this number is not greater than the number of defined abilities for the object.

```
Ability 1 Chance            := 5
```

This determines what percent (measured in tenths of a percent) of the objects placed on a map will have the particular ability. The amount of percentage leftover from the total of the Chances (subtracted from 1000) of all possible abilities for the objects is the percentage of objects that will have no ability. In this case, 0.5% of all Normal Planets will have Ability 1.

```
Ability 1 Type              := Ancient Ruins
```

This tells SE4 what type of ability the ability will be.

```
Ability 1 Val 1             := 1
Ability 1 Val 2             := 0
```

These tell SE4 what precisely the ability does. They have different effects for different abilities.

Return to Table of Contents

Chapter 10: Events

Events are defined in Events.txt. There isn't much to them. All the possible types of events are:

```
Replacement Symbols used for Messages:
[%SystemName]           - Target System Name
[%SectorName]           - Target Sector Name (Number)
[%SourceEmperorName]    - Source Emperor's Name (includes Title)
[%SourceEmpireName]     - Source Empire Name (includes Empire Type)
[%VehicleName]          - Vehicle Name
[%VehicleSize]          - Vehicle Size
[%PlanetName]           - Planet Name
[%DesignName]           - Design Name
[%TechName]             - Tech Area Name
[%TreatyName]           - Political Treaty
[%FacilityName]         - Facility Name
[%StarName]             - Star Name
[%WarpPointName]        - Warp Point Name
[%ActualAmount]         - Actual Effect Amount (Can't be used for Start
Message)
```

This notation (also used in some AI files) can be used for customized event messages.

```
Type                        := Ship - Damage
```

The event types are as follows:

Ship - Damage

This cause Effect Amount points of damage to a ship. It bypasses shielding.

Ship - Lose Movement,

Effect Amount movement points will be lost for one turn for the affected ship. If a ship in a fleet is hit, the entire fleet will be slowed down effectively (the other ships will not lose any movement points, but fleets move at the speed of the slowest ship).

Ship - Lose Supply

The affected ship will lose Effect Amount supplies.

Ship - Experience Change,

The ship's experience level will change by Effect Amount.

Ship - Cargo Damage

Effect Amount damage will be done to the cargo aboard the affected ship.

Ship - Moved

The affected ship will be moved to a random sector of a random system in the galaxy. Effect Amount is meaningless for this ability.

Planet - Conditions Change

The conditions of the affected planet will change by Effect Amount points. Basically, conditions are stored as a decimal ranging from 0.00 to 1.50. This ability seems to drop the conditions value by Effect Amount / 10 points. So, with an Amount of 10, the conditions get dropped by 1.00 points.

Planet - Value Change

The value of the affected planet will be changed by Effect Amount percent (added or subtracted, not multiplied).

Planet - Population Change

The target planet's population will be changed by Effect Amount M people.

Planet - Population Anger Change,

The anger of the target planet will be altered by Effect Amount points. Positive makes them more angry, negative less angry (more happy).

```
Planet - Population Riot
```

The target planet's anger will be maximized, causing an instant riot. Effect Amount has no effect.

```
Planet - Population Rebel
```

The planet will rebel, either forming a new empire (for event or intel project), or (only for an intel project) will join the empire that performed the project.

```
Planet - Cargo Damage
```

Effect Amount damage will be done to the cargo stored on a planet.

```
Planet - Facility Damage
```

Effect Amount facilities on the planet will be destroyed.

```
Points - Change
```

Effect Amount resources (of each type, minerals, organics and radioactives) will be removed or added to the target empire's storage.

```
Research - Delete Project
```

A research project will be deleted from the research queue, losing all points invested in it.

```
Intel - Delete Project
```

An intelligence project will be deleted from the intelligence queue, losing all points invested in it.

```
Planet - Created
```

A planet will be created in a random sector in the galaxy. Effect Amount has no effect.

```
Planet - Destroyed
```

A random planet will be destroyed in a random system in the galaxy. Effect Amount has no effect.

```
Star - Created
```

A star will be created in a random system in the galaxy, presumably in a system without any stars. Effect Amount has no effect.

```
Star - Destroyed
```

A star will be destroyed in a random system in the galaxy. Effect Amount has no effect.

```
Warp Point - Closed
```

A warp point will be closed in a random system in the galaxy. Effect Amount has no effect.

```
Warp Point - Opened
```

A warp point will be opened in a random system in the galaxy. Effect Amount has no effect.

```
Planet - Plague
```

A level Effect Amount plague will be created on a random inhabited planet in the galaxy. You can not change the effects of the plagues, just use a hard-coded level 1-5 plague.

```
Planet - Plague Cured
```

A plague of up to level Effect Amount will be cured on a random planet. Only planets that actually have a plague will be selected by a random event of this type.

```
Severity                    := Low
```

This limits what types of events can occur in a game. The possible severities are Low, Medium, High, Catastrophic. At game start, the host can select one of these options for event severity. No events of a higher severity than the option selected can occur in the game. For example, if the severity is set to Medium, no High or Catastrophic events can occur.

```
Effect Amount               := 50
```

What this does is dependant upon the type of event. For example, for a Planet - Population Change event, this determines by how much the population changes. For a Planet - Plague event, it determines the level of plague that affects the target planet (max level 5).

```
Message To                  := Owner
```

This determines who receives notification when an event occurs. It can be set to None, Owner, Sector, System, All. None means no-one will receive notification. Owner means that only the owner of the affected object will be notified. Sector means that all empires present in the sector where the targeted object is located will be notified. System means that all empires present in the system where the targeted object is located will be notified. All means that all empires, regardless of their location, will be notified of the occurrence of the event.

```
Num Messages                := 1
```

This tells SE4 how many messages to display when the effects of the event happen. It is used when the body of the message is too long to display in one screen.

```
Message Title 1             := Ion Storm
Message 1                   := An ion storm has damaged ship
[%VehicleName] in the [%SystemName] system.
Picture                     := ShipDamaged
```

These fields determine the title of the message given when the effects of the event occur (not when it is first called for), the body of said message, and the image displayed next to that message.

```
Time Till Completion        := 0
```

This is the delay (in turns) between when the event is called for, and when it's effects actually happen. After the Time Till Completion occurs, the effects of the event happen, and the Message is displayed.

```
Num Start Messages          := 0
```

This message, if any, is displayed when an event is first called for.

Return to Table of Contents

Chapter 11: Intel Projects

Intelligence Projects are nearly identical to Events, except for a few minor differences. You can use nearly all (if not all) of the event abilities as Intel Projects. The reverse is generally not true though. Check out Chapter 10: Events for descriptions of some of the effect abilities (those that are present in the Events file and that might need some explanations).

```
Group                       := Ship Sabotage
```

The Group is merely an organizational tool. It tells SE4 where to place the project in the list of available projects for an empire to perform.

```
Cost                        := 10000
```

This is how many intelligence points it takes to complete the project. This value not only determines how long it takes to complete the project, but also how difficult to defeat by enemy counter-intel.

For counter-intel type projects, the cost is the limit on the number of points that may be spent on the project (and thus the maximum points used for defense).

```
Num Source Messages        := 1
Num Target Messages        := 1
```

With Intel Projects, the empire performing the project and the target empire receive different messages. The Source Message goes to the performing empire, and the Target Message goes to the targeted empire(s).

```
Source Picture             := IntelSabotageByUs
Target Picture             := ShipDamaged
```

As with Messages, the picture displayed alongside the messages can be different for the performing and target empires.

```
Number of Tech Req         := 1
Tech Area Req 1            := Applied Intelligence
Tech Level Req 1           := 1
```

These lines determine what technologies are required before an empire may perform the Intel Project.

Return to Table of Contents

Chapter 12: Formations

The graphical outlines of the formations found at the beginning of Formations.txt are never referenced by SE4. They are only there as a mechanism to aid in designing the formation, and for viewing what it will look like outside of the game.

```
Arrowhead
 00000000011111111111
 12345678901234567890
01
02
03
04
05
06
07
08
09         L
10        1 2
```

```
11       3    4
12      5      6
13     7    F    8
14    9    G H    A
15   B    I    J    C
16 D    K      L    E
17
18
19
```

This is the layout of the Arrowhead Formation. Remember, this picture is not at all necessary in the Formations.txt file.

```
Name                        := Arrowhead
Description                 := Formation with...
```

These are the Name of the formation and the Description shown when viewing the formation in-game.

```
Leader Position Xpos        := 10
Leader Position Ypos        := 9
```

These define the position of the Leader ship of the Formation. The grid used is a 19 x 19 square, with 1 in the top left corner for each axis. See the above graphic.

```
Leader Design Type          := Any
```

This will have only have an effect if the leader of the fleet out of combat is set as a non-combat ship (which will break formation and run for the corners). The leader of the fleet will then be a ship of this particular design (or the 2nd ship in the fleet, in order of construction, if set to any).

```
Number of Positions         := 21
```

This is the number of possible positions in the fleet. If there are more combat ships in the fleet than this number, the extras will be placed more or less randomly on the combat map, and not follow the fleet leader around. There can be up to 100 ships designated in the formation.

```
Position 1 Xpos             := 9
Position 1 Ypos             := 10
```

This is used to tell SE4 where to place the next (non-leader) ship in the fleet. The grid used is a 19 x 19 square, with 1 in the top left corner for each axis. See the above graphic.

```
Position 1 Type            := Any
```

This can be used to tell SE4 to place a specific design type in this position. This could, for example, be used to have Missile Ships be placed in the rear of the fleet, and Beam Weapon Ships up in front.

Return to Table of Contents

Chapter 13: Happiness

The Happiness Level of a colony is measured by the percent of the population that is angry. An event with a positive value increases this anger percent, thereby lowering happiness. An event with a negative value lowers the anger percent, thereby increasing happiness. The anger percents for each Happiness Level are as follows:

```
Population Happiness Levels:
Pop Anger Pct      Description     Res Mod
75 - 100           Rioting         0%
60 - 74            Angry           60%
45 - 59            Unhappy         80%
30 - 44            Indifferent     100%
15 - 29            Happy           120%
0 - 14             Jubiliant       140%
```

Now, we shall take a look at each possible event that can affect happiness. All values are in tenths of a percent.

```
Max Positive Anger Change        := 200
```

This is the maximum amount that the anger level of a planet can change in a positive direction in 1 turn.

```
Max Negative Anger Change        := -200
```

This is the maximum amount that the anger level of a planet can change in a negative direction in 1 turn.

```
Homeworld Lost                   := 100
```

This is the amount that anger is changed by when a planet designated as a Homeworld is lost to an enemy empire.

```
Any Planet Lost                  := 50
```

This is the amount that anger is changed by when any non-Homeworld planet is lost to an enemy empire.

```
Any Planet Colonized                := -10
```

When the empire colonizes a planet, the anger of all of it's planets is changed by this value.

```
Any Our Planet Captured             := 50
```

When a planet owned by the empire is captured by an enemy empire, all of it's other planets have their anger changed by this value.

```
Any Enemy Planet Captured           := -30
```

When the empire captures an enemy planet, it's planets' anger levels are changed by this value.

```
Any Ship Lost                       := 1
```

When the empire loses a ship, whether through combat or natural phenomena, the anger levels of all of it's planets are changed by this value.

```
Any Ship Constructed                := 0
```

When the empire constructs a ship, all of the anger levels of it's planets are changed by this value.

```
New Treaty X                      := 100
```

The planets in the empire have their anger levels changed by this value when the empire establishes a new treaty or a war. It is the same process (not value) for all treaty types.

```
Battle in System - Win              := -20
```

If a battle occurs in the same system as a planet, and the empire wins, the planet's anger level is changed by this value.

```
Battle in System - Loss             := 20
```

If a battle occurs in the same system as a planet, and the empire loses, the planet's anger level is changed by this value.

```
Battle in System - Stalemate        := 0
```

If a battle occurs in the same system as a planet, and the battle results in a stalemate (no one wins), the planet's anger level is changed by this value.

```
Battle in Sector - Win          := -50
```

If a battle occurs in the same sector as a planet, and the empire wins, the planet's anger level is changed by this value. This value supercedes the Battle in System value.

```
Battle in Sector - Loss         := 50
```

If a battle occurs in the same sector as a planet, and the empire loses, the planet's anger level is changed by this value. This value supercedes the Battle in System value.

```
Battle in Sector - Stalemate    := 10
```

If a battle occurs in the same sector as a planet, and the battle results in a stalemate (no one wins), the planet's anger level is changed by this value. This value supercedes the Battle in System value.

```
Enemy Ship in System            := 5
```

When an enemy ship is present in the same system as a planet, that planet's anger level is changed by this value.

```
Enemy Ship in Sector            := 8
```

When an enemy ship is present in the same sector as a planet, that planet's anger level is changed by this value. This value supercedes the Enemy Ship in System value.

```
Our Ship in Sector              := -10
```

When a ship owned by the empire or an ally is present in the same sector as a planet, that planet's anger level is changed by this value. This value supercedes the Enemy Ship in System value.

```
Our Ship in System              := -3
```

When a ship owned by the empire or an ally is present in the same system as a planet, that planet's anger level is changed by this value. This value supercedes the Enemy Ship in System value.

```
Enemy Troops on Planet          := 200
```

Each enemy troop remaining on a planet at the end of the turn changes the planet's anger level by this value.

```
Our Troops on Planet            := -2
```

Each troop garrisoned on the planet changes the planet's anger level by this value.

```
1M Population Killed              := 1
```

Each 1M population killed on a planet, whether from an event or enemy bombardment, changes the planet's anger level by this amount.

```
Ship Lost in System               := 2
```

When the empire loses a ship, whether through combat or natural phenomena, the anger levels of all of it's planets in the same system as the lost ship are changed by this value.

```
Ship Constructed                  := -5
```

When the empire constructs a ship, the anger level of the planets in the same sector as the constructed ship are changed by this value.

```
Facility Constructed              := -5
```

When a planet constructs a facility, it's anger level is changed by this value.

```
Planet Plagued                    := 200
```

When a planet is affected by a plague, it's anger level is changed by this value.

```
Natural Decrease                  := -20
```

This is a slight ambient affect that changes the anger level of each planet in the empire that has population native to that empire, either each turn, or every X turns.

```
Natural Decrease for Other Races := 20
```

This is a slight ambient affect that changes the anger level of each planet in the empire that has population that is not native to that empire (whether traded for, subjugated, or acquired through any other means), either each turn, or every X turns.

Return to Table of Contents

Chapter 14: Racial Traits

Racial traits affect the empire as a whole. Here is an example trait:

```
Name                    := Organic Manipulation
Description             := Gains access to the Organic Technology Tree.
Pic Num                 := 0
General Type            := Advantage
```

```
Cost                    := 1500
Trait Type              := Tech Area
Value 1                 := 5
Value 2                 := 0
Required Trait 1        := None
Required Trait 2        := None
Required Trait 3        := None
Restricted Trait 1      := None
Restricted Trait 2      := None
Restricted Trait 3      := None

Pic Num                 := 0
```

This field is meaningless. There is never a picture displayed next to racial traits.

```
General Type            := Advantage
```

This field is also meaningless. When Traits are displayed in-game, they are displayed in the order that they are listed in the file. They cannot be sorted.

```
Cost                    := 1500
```

This value is subtracted from the remaining racial points when designing an empire. If negative (for a racial penalty trait), the points increase the racial point balance.

```
Trait Type              := Tech Area
Value 1                 := 5
Value 2                 := 0
```

The Trait Type determines what the racial trait does. By setting it to Tech Area, you can create a racial tech tree. The Value 1 is the number used for the Racial Area setting when designing racial tech areas.

```
Required Trait 1        := None
Restricted Trait 1      := None
```

These settings are non-functional. In theory, when designing an empire in-game, you would be required to select a Required Trait in order to select this trait. You would not be able to take this trait if you had selected a Restricted Trait.

There are many Trait Types that will work in this file. Here is a list of all of the abilities that are used in the default files:

```
Supply Cost
No Plagues
No Spaceports
Luck
Vehicle Speed
Galaxy Seen
Planet Storage Space
Planetary SY Rate
Tech Area
Population Emotionless
```

The following list is a list of other abilities that are listed in the SE4 executable that may or may not work:

```
Troops Bonus
Fighter Bonus
Ship Bonus
Mineral Production
Mineral Storage
Organics Production
Organics Storage
Radioactives Production
Radioactives Storage
Research Production
Intelligence Production
Trade
Ground Combat
Space Combat
Maintenance Cost
Ship Attack
Ship Defense
```

Return to Table of Contents

Chapter 15: Miscellaneous Files

Return to Table of Contents

Chapter 16: Abilities

In this chapter, we shall take an in-depth look at the abilities available in the game. Check Data\Abilities.txt for a basic listing. Contributions in this chapter have been made by Suicide Junkie.

All abilities list what files the ability is valid in (i.e.: what type of objects it will function on). Abilities used in invalid files will basically be "dummy abilities". They do nothing but add their description to the bulleted list of abilities for that object.

```
Warp Point - Turbulence
  Value1 = Amount of damage done to objects moving through this warp point. (Normal Damage)
  Value2 =
```

Valid in StellarAbilityTypes.txt

When a vehicle enters the sector containing a warp point with this ability, it will take the amount of damage specified by Value1. The damage is equivalent to a mine hit of the same strength, one for each ship. Shields do not protect against this type of damage.

```
Star - Unstable
  Value1 = Chance that sun will explode each year.
  Value2 =
```

Valid in StellarAbilityTypes.txt

This ability actually does nothing. "Unstable" stars have no greater chance of exploding than normal stars, and the Supernova type events can affect any star with an equal chance.

Because this ability does nothing, it is a good candidate to use for description abilities in components, facilities, etc. You can assign this ability (or any "dummy" ability) to the component, and its description is listed at the bottom, with other ability descriptions.

```
Sector - Sight Obscuration
  Value1 = Level of obscuration. (Obscures all types) (Units cannot use this ability)
  Value2 =
```

Valid in Components.txt, Facility.txt, StellarAbilityTypes.txt, SystemTypes.txt, VehicleSize.txt

This ability can be placed on stellar phenomena, components, ship hulls or facilities. All objects in the sector will be given an "always-on" cloaking ability of the amount specified by Value1. Even planets and stars can be made invisible (and thus unattackable) by this ability. Sensors of an equal or higher ability amount can penetrate this cloaking effect.

```
Sector - Sensor Interference
  Value1 = Modifier subtracted from to-hit rolls
  Value2 =
```

Valid in Components.txt, Facility.txt, StellarAbilityTypes.txt, SystemTypes.txt, VehicleSize.txt

Any non-seeker weapon fired in this sector has Value1 subtracted from its chance to hit. It is the same as an ECM generator built into every ship.

```
Sector - Shield Disruption
  Value1 = Amount of shields lost during combat.
  Value2 =
```

Valid in Components.txt, Facility.txt, StellarAbilityTypes.txt, SystemTypes.txt, VehicleSize.txt

Each ship in this sector will have Value1 fewer maximum shield points while in combat. Regeneration is unaffected, and the starting shield points are set at the lowered maximum. Any ships designed with fewer than Value1 shield points will be completely unshielded.

You can also use a negative value here to add shield points to all vehicles in the sector. Shields will even be added to ships that have no shield generating components.

```
Sector - Damage
  Value1 = Amount of damage per turn to all objects in sector.
  Value2 =
```

Valid in Components.txt, Facility.txt, StellarAbilityTypes.txt, SystemTypes.txt, VehicleSize.txt

All vehicles that enter the sector will be damaged by Value1 points. Note that there is only a roughly 66% chance that this ability will trigger. Also, the affects of multiple objects in the sector with this ability stack.

```
Resource Generation - Minerals
  Value1 = Amount of minerals generated per turn.
  Value2 =
```

Valid in Facility.txt

The base amount of minerals that a facility will produce. This is modified by planet population, happiness, planet value, and racial aptitude.

```
Resource Generation - Organics
  Value1 = Amount of organics generated per turn.
  Value2 =
```

Valid in Facility.txt

The base amount of organics that a facility will produce. This is modified by planet population, happiness, planet value, and racial aptitude.

```
Resource Generation - Radioactives
  Value1 = Amount of radioactives generated per turn.
  Value2 =
```

Valid in Facility.txt

The base amount of radioactives that a facility will produce. This is modified by planet population, happiness, planet value, and racial aptitude.

```
Point Generation - Research
  Value1 = Amount of research points generated per turn.
  Value2 =
```

Valid in Facility.txt

The base amount of research points that a facility will produce. This is modified by planet population, happiness, and racial aptitude.

```
Point Generation - Intelligence
  Value1 = Amount of intelligence points generated per turn.
  Value2 =
```

Valid in Facility.txt

The base amount of intel points that a facility will produce. This is modified by planet population, happiness, and racial aptitude.

```
Spaceport
  Value1 =
  Value2 =
```

Valid in Facility.txt

A facility with this ability allows the resources, intel and research points generated in a system to be used by the empire. Races with the "Natural Merchants" trait do not need to build facilities with this ability.

```
Palace
  Value1 =
  Value2 =
```

Valid in no file

This has no effect as of yet.

```
Supply Generation
  Value1 =
  Value2 =
```

Facility.txt

When on a facility, any friendly ship or unit that enters the sector will be automatically resupplied to full supply level.

```
Planet - Change Minerals Value
  Value1 = Percentage the value of this planet is changed each turn.
  Value2 =
```

Valid in Facility.txt

See Change Radioactives Value entry below.

```
Planet - Change Organics Value
  Value1 = Percentage the value of this planet is changed each turn.
```

```
  Value2 =
```

Valid in Facility.txt

See Change Radioactives Value entry below.

```
Planet - Change Radioactives Value
  Value1 = Percentage the value of this planet is changed each turn.
  Value2 =
```

Valid in Facility.txt

This ability takes effect on the "0" month of each year (not each turn). In a finite resources game, the total resources are multiplied by (100% + Value1). In a infinite resources game, Value1 is added to the % value of the planet.

```
Planet - Change Conditions
  Value1 = Percentage the conditions of this planet is changed each
turn.
  Value2 =
```

Valid in Facility.txt

This ability takes effect on the "0" month of each year (not each turn).

```
Planet - Change Population Happiness
  Value1 = Percentage the happiness of this planet's population is
improved each turn.
  Value2 =
```

Valid in Facility.txt

This ability raises or lowers the anger level of the planet each turn by Value1. A positive value raises anger levels, thus making the population less happy. A negative value lowers anger levels, thus making the population happier. Different facilities with this ability seem to stack.

```
Planet - Change Ground Defense
  Value1 = Percentage modifier to ground combat on this planet.
  Value2 =
```

Valid in no file

This has no effect as of yet.

```
Planet - Shield Generation
  Value1 = Amount of shields generated.
  Value2 =
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

This ability adds Value1 normal shield points to a planet or vehicle.

```
Shield Generation
  Value1 = Amount of shields generated.
  Value2 =
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

This ability adds Value1 normal shield points to a planet or vehicle.

```
Phased Shield Generation
  Value1 = Amount of phased shields generated.
  Value2 =
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

This ability adds Value1 phased shield points to a planet or vehicle. If one component or facility generates 1 normal (non-phased) shield point, then all shields on the ship or planet are considered non-phased. If that component is destroyed, then the shields will become phased if all functional components/facilities generate phased shields only.

```
Component Repair
  Value1 = Number of components repaired per turn.
  Value2 =
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

At the end of each turn, this component will repair Value1 other components. The repairs can be to one ship or many, and multiple repair components can repair the same vehicle. Which particular vehicle and components are repaired depends on the Repair priorities that are defined

in the empire options window.

```
Cargo Storage
  Value1 = Number of cargo spaces generated.
  Value2 =
```

Valid in Components.txt, Facility.txt, StellarAbilityTypes.txt, VehicleSize.txt

This vehicle can store units with a combined total mass less than or equal to Value1. Multiple components with this ability form one continuous cargo bay, so two components can hold a unit bigger than any one individual component could.

```
Drop Troops
  Value1 = Number of troops which can be dropped to attack a planet.
  Value2 =
```

Valid in no file

This has no effect as of yet.

```
Launch/Recover Fighters
  Value1 = Amount of fighters that can be launched per combat turn.
  Value2 = Amount of fighters that can be launched per game turn.
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

This component adds the ability to launch fighters from a vehicle. Value1 fighters of any size can be launched during each round of combat, and Value2 fighters of any size can be launched during a game month. Any number of fighters may be recovered, up to the capacity of the vehicles cargo space. Planets can launch an unlimited number of fighters per turn, so placing this ability on a facility is meaningless.

```
Lay Mines
  Value1 = Amount of mines that can be launched per combat turn.
  Value2 = Amount of mines that can be launched per game turn.
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

This component adds the ability to launch mines from a vehicle. Value1 mines of any size can be launched during each round of combat, and Value2 mines of any size can be launched during a

game month. Mines can never be recovered from space. Mines launched during combat are visible and vulnerable to weapons fire until the end of combat. Any mines not destroyed will deploy normally and become cloaked. Planets can launch an unlimited number of mines per turn, so placing this ability on a facility is meaningless.

```
Multiplex Tracking
  Value1 = Number of ships that can be tracked and fired on.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

This ability allows a ship to fire its main guns at separate targets during a single combat round. Automatically fired Point defense will not count towards the number of targets fired on, but manually firing point defense weapons in tactical combat will.

```
Combat To Hit Offense Plus
  Value1 = Percent modifier for combat offense.
          (Increases the chance that this ship will hit another ship).
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

This ability adds Value1 % to the chance of this vehicle hitting its target with direct-fire weapons. The modifier adds linearly, so 50% + 30% = 80%

Several bugs, discovered by Aiken, interfere when you add this ability to a weapon platform hull. The bonus from each weapon platform will stack. If you add a +10% bonus to hit and have 5 weapon platforms, the total bonus from this ability will be +50%, rather than the +10% you would expect. Further, if you add some troops with a +10% bonus to hit to the planetary cargo, that +10% will add to the to hit bonus of the weapon platforms, for a total of +60%. Only one troop bonus applies. If you add several fighters, the total to hit bonus from each fighter is added to the weapon platforms' chance to hit. If you add 5 fighters with a 50% bonus, the total to hit bonus applied to the weapon platforms will be 60 + 250, for +310% bonus.

```
Combat To Hit Defense Plus
  Value1 = Percent modifier for combat defense.
          (Decreases the chance that this ship will be hit by another
ship).
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

This ability subtracts Value1 % from the chance of enemy direct-fire weapons hitting this vessel. The modifier subtracts linearly, so 50% - 30% = 20%

```
Mine Sweeping
  Value1 = Number of mines that can be swept.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

When this vehicle enters a sector containing mines, Value1 mines will be disabled before any damage is done to the fleet. Cloaked ships with minesweepers on them can not sweep mines.

```
Medical Bay
  Value1 = Plague level that can be cured.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

If this vehicle enters the same sector as a planet that has plague conditions of level Value1 or less, it will cure the plague, and return the conditions to normal.

```
Movement Bonus
  Value1 = Added movement for ship if using all of these types of
engines.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

After all other propulsion modifiers are calculated, a ship that has at least one movement point will have Value1 added to its speed. The smallest Value1 of any component on the ship will be used. (All engines do not have to be the same, but the weakest engine's ability will be used)

```
Emissive Armor
  Value1 = Damage armor must take before being destroyed.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

When a weapon hits a ship with this ability on it, Value1 is subtracted from the remaining weapon strength before the damage is applied. The weapon does not actually have to hit the

component with this ability on it. Multiple components with this ability do not stack. A weapon with either the Skips Armor or Skips Shields And Armor damage type will never trigger this ability.

```
Shield Regeneration
  Value1 = Number of shields regenerated per combat turn.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

At the end of each combat round, this vehicle will add Value1 points to its shield strength. Points beyond the maximum remaining shield strength are lost.

```
Master Computer
  Value1 =
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

Vehicles with this component do not require Bridge, crew quarters or life support. If the ability is given to the vehicle type itself, the vehicle will still require B/CQ/LS, but still gains the following abilities:
The vehicle will not lose movement points if B/CQ/LS are destroyed.
The vehicle is immune to the Crew Conversion damage type (Allegiance Subverter).

```
Cloak Level
  Value1 = Type of sight obscuration.
  Value2 = Level of obscuration.
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

Provides Value2 cloaking of the Value1 type. There are 5 types of cloaking, Passive EM, Active EM, Gravitic, Psychic and Temporal. Level X scanners defeat level X or lower cloaking. So, a level 5 cloak is needed to keep ships hidden against level 4 scanners. All vehicles and planets get a built-in level 1 EM Passive scanner and a level 1 EM Active scanner, so a level 1 EM cloak of either (or both) type is pointless.

```
Sensor Level
  Value1 = Type of sight ability.
  Value2 = Level of sight ability.
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

This ability is used to detect cloaked ships. Value1 defines the type of cloaking ability that can be defeated. It can be EM Active, EM Passive, Gravitic, Psychic or Temporal. Value2 is the level of Value1 cloaking that can be defeated. Value2 scanning ability will defeat Value2 cloaking ability. All ships receive level 1 in EM Active and level 1 EM Passive scanning by default, so level 1 cloaking in either (or both) EM type is wholly ineffective.

```
Emergency Resupply
  Value1 = Amount of resupply given to ship on use.
  Value2 =
```

Valid in Components.txt

When a vehicle is given an order to use this component, the vehicle will gain Value1 supplies. Any supplies beyond the maximum capacity of the vehicle are lost. When used this component will be destroyed. This component requires a spaceyard to be repaired. Repair bays will not work on this component. You can give this ability to components that you want to only be repairable at a Space Yard with Value1 set to 0.

```
Emergency Energy
  Value1 = Amount of movement given to ship on use.
  Value2 =
```

Valid in Components.txt

When a vehicle is given an order to use this component, the vehicle will gain Value1 more movement points. Movement points beyond the maximum speed of the ship are NOT lost. In simultaneous games, the number of days between moves will decrease. Repair bays will not work on this component. You can give this ability to components that you want to only be repairable at a Space Yard with Value1 set to 0.

```
Long Range Scanner
  Value1 = Distance away in sectors enemy ship can be scanned.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

The cargo and component status of enemy vehicles within Value1 sectors of this component can be viewed as if the ships were owned by this empire. Invisible (cloaked) enemy vehicles, and those with Scanner Jammer ability cannot be viewed.

```
Open Warp Point Distance
  Value1 = Distance to system (in light years) WP can be opened to.
  Value2 =
```

Valid in Components.txt

This ability allows the ship to create new warp points, to a maximum distance of Value1 light years (where each square on the galaxy map is 10 light years).

```
Create Planet Size
  Value1 = Size of planet that can be created.
  Value2 =
```

Valid in Components.txt

This will create a planet up to size Value1. The maximum size of planet is based on the position of the planets in PlanetSize.txt. So, a Value1 of 3 can create up to the 3rd listed planet in PlanetSize.txt (medium in unmodded SE4).

```
Destroy Planet Size
  Value1 = Size of planet that can be destroyed.
  Value2 =
```

Valid in Components.txt

This will destroy a planet up to size Value1 and create an asteroid field of the same size in its place. The maximum size of planet is based on the position of the planets in PlanetSize.txt. So, a Value1 of 3 can create up to the 3rd listed planet in PlanetSize.txt (medium in unmodded SE4).

```
Boarding Attack
  Value1 = Attack strength in ship capture.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

This ship can be ordered to board enemy vehicles that have zero shield points remaining and are one combat square away. If the sum total of Boarding Attack Value1 on this ship exceeds the sum total of Boarding Defense Value1 on the target vessel, then the target vessel becomes the property of this empire. If the target vessel had an intact self destruct ability, then both ships are destroyed. Regardless of the success of the operation, this vessel's shields will be set to zero. If the target vessel is captured, its weapons' reload time will be increased by the amount specified in settings.txt.

```
Boarding Defense
  Value1 = Defense strength in ship capture.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

See Boarding Attack. Value1 points of boarding defense are added to this vehicle. Crew Quarters have a hard-coded, built-in boarding defense of 4 points.

```
Standard Ship Movement
  Value1 = Number of movement points generated.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

Value1 thrust points are added to this vehicle. The total thrust is divided by the "Engines per Move" setting in VehicleSize.txt and rounded down to get the actual movement speed of the vehicle.

This is the only ability that causes a component to contribute to the amount of supplies used for each sector of movement. Basically, sum up the Supply Amount Used value for every component on the ship that provides at least one Standard Ship Movement point (not bonus move, or extra movement generation, or combat movement). This value is how many supplies are used each time the ship moves one sector on the system map. As multiple sectors can be traversed in a turn, the supplies used in a turn of movement will be the sum times sectors moved.

Note that bases can never have system movement points. This ability does not work on bases. In the design screen, movement points will be calculated. But once the base is built, it will not have any movement points.

```
Ship Bridge
  Value1 =
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

Vehicles without a Master Computer ability require Exactly One (1) component with this ability. If a vehicle has no master computer, and it loses its bridge ability for any reason, it will suffer a penalty of half its movement speed.

```
Ship Auxiliary Control
  Value1 =
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

During combat, vehicles with this ability can lose their bridge ability without suffering a movement penalty. Outside of combat, the penalty still applies.

```
Ship Life Support
  Value1 =
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

Vehicles without a Master Computer ability require components with this ability. The number required for a legal design is specified in VehicleSize.txt. If a vehicle has no master computer, and it loses ALL of its life support ability for any reason, it will lose three-quarters of its movement speed.

```
Ship Crew Quarters
  Value1 =
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

Vehicles without a Master Computer ability require components with this ability. The number required for a legal design is specified in VehicleSize.txt. If a vehicle has no master computer, and it loses ALL of its Crew Quarters ability for any reason, it will lose half of its movement speed.

```
Scanner Jammer
  Value1 =
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

This prevents the use of the Long Range Scanner ability in this ship.

```
Quantum Reactor
  Value1 =
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

This vehicle gains an infinite amount of supplies. When placed in a fleet of other vehicles, it will restore their supplies to maximum at the end of the turn. Bases essentially have a built-in hard-coded Quantum Reactor ability.

```
Supply Storage
  Value1 = Amount of supplies that can be stored.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

This vehicle can store an additional Value1 points of supplies.

```
Space Yard
  Value1 = Resource Type used for construction
  Value2 = Amount of resources which can be used each turn.
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

Value1 is one of: Minerals, Organics, Radioactives. Value2 is the build rate of the component in the resource specified by Value1. For typical use, three copies of this ability are used on a single component, one for each resource. Only one component with spaceyard abilities can be included on a single vehicle. A second facility containing this ability cannot be constructed, although an upgrade is legal.

```
Resource Storage - Mineral
  Value1 = Added amount of resources an empire can store.
  Value2 =
```

Valid in Facility.txt

This empire can store an additional Value1 minerals.

```
Resource Storage - Organics
```

```
  Value1 = Added amount of resources an empire can store.
  Value2 =
```

Valid in Facility.txt

This empire can store an additional Value1 organics.

```
Resource Storage - Radioactives
  Value1 = Added amount of resources an empire can store.
  Value2 =
```

Valid in Facility.txt

This empire can store an additional Value1 radioactives.

```
Resource Gen Modifier Planet - Minerals
  Value1 = Percentage change of resource generation for entire planet
(+/- percentage).
  Value2 =
```

Valid in Facility.txt

Mineral resource production rates on this planet have Value1 % added to them. This modifier adds, so 80% - 30% = 50%

```
Resource Gen Modifier Planet - Organics
  Value1 = Percentage change of resource generation for entire planet
(+/- percentage).
  Value2 =
```

Valid in Facility.txt

Organic resource production rates on this planet have Value1 % added to them. This modifier adds, so 80% - 30% = 50%

```
Resource Gen Modifier Planet - Radioactives
  Value1 = Percentage change of resource generation for entire planet
(+/- percentage).
  Value2 =
```

Valid in Facility.txt

Radioactives resource production rates on this planet have Value1 % added to them. This modifier adds, so 80% - 30% = 50%

```
Resource Gen Modifier System - Minerals
  Value1 = Percentage change of resource generation for entire system
(+/- percentage).
  Value2 =
```

Valid in Facility.txt

Mineral resource production rates on all planets in this system have Value1 % added to them. This modifier adds, so 80% - 30% = 50%. This modifier does not show up on the planet reports, but does make a difference in the budget window.

```
Resource Gen Modifier System - Organics
  Value1 = Percentage change of resource generation for entire system
(+/- percentage).
  Value2 =
```

Valid in Facility.txt

Organic resource production rates on all planets in this system have Value1 % added to them. This modifier adds, so 80% - 30% = 50%. This modifier does not show up on the planet reports, but does make a difference in the budget window.

```
Resource Gen Modifier System - Radioactives
  Value1 = Percentage change of resource generation for entire system
(+/- percentage).
  Value2 =
```

Valid in Facility.txt

Radioactives resource production rates on all planets in this system have Value1 % added to them. This modifier adds, so 80% - 30% = 50%. This modifier does not show up on the planet reports, but does make a difference in the budget window.

```
Planet Point Generation Modifier - Research
  Value1 = Percentage change of point generation for entire planet
```

```
(+/- percentage).
  Value2 =
```

Valid in Facility.txt

Research rates on this planet have Value1 % added to them. This modifier adds, so 80% - 30% = 50%

```
Planet Point Generation Modifier - Intelligence
  Value1 = Percentage change of point generation for entire planet
(+/- percentage).
  Value2 =
```

Valid in Facility.txt

Intel production on this planet have Value1 % added to them. This modifier adds, so 80% - 30% = 50%

```
System Point Generation Modifier - Research
  Value1 = Percentage change of point generation for entire system
(+/- percentage).
  Value2 =
```

Valid in Facility.txt

Research rates on all planets in this system have Value1 % added to them. This modifier adds, so 80% - 30% = 50%. This modifier does not show up on the planet reports, but does make a difference in the research window.

```
System Point Generation Modifier - Intelligence
  Value1 = Percentage change of point generation for entire system
(+/- percentage).
  Value2 =
```

Valid in Facility.txt

Intel production on all planets in this system have Value1 % added to them. This modifier adds, so 80% - 30% = 50%. This modifier does not show up on the planet reports, but does make a difference in the intel window.

```
Combat Modifier - System
  Value1 = Percentage combat modifier for entire system (+/-
percentage).
  Value2 =
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

This ability adds a Combat Sensors effect to all ships owned by this empire in this system. The chance to hit with a weapon is increased by Value1.

```
Damage Modifier - System
  Value1 = Percentage damage modifier for entire system (+/-
percentage).
  Value2 =
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

Damage done by all weapons fired by this empire in this system in increased by Value1%

```
Planet Value Change - System
  Value1 = Percentage change in planet value for entire system (+/-
percentage) per year.
  Value2 =
```

Valid in Facility.txt

On each "0" month, all planets in this system that are controlled by this empire will gain resource value. In a finite resources game, the increase will be Value1 % of the total resources of each type. In a infinite resources game, the Value1 percentage points will be added to the planet value.

```
Planet Conditions Change - System
  Value1 = Percentage change in planet conditions for entire system
(+/- percentage) per year.
  Value2 =
```

Valid in Facility.txt

On each "0" month, all planets in this system that are controlled by this empire will gain conditions value. Conditions are stored in a variable that ranges from 0.00 to 1.50. So, Value1 is added to the planet's conditions as a decimal each time.

```
Change Bad Event Chance - System
  Value1 = Percentage change in chance for bad event for entire system
(+/- percentage).
  Value2 =
```

Valid in Facility.txt

This ability may or may not be functional.

```
Change Bad Intelligence Chance - System
  Value1 = Percentage change in chance for bad intel event for entire
system (+/- percentage).
  Value2 =
```

Valid in Facility.txt

This ability may or may not be functional.

```
Change Population Happiness - System
  Value1 = Percentage change in population happiness for entire system
(+/- percentage) each turn.
  Value2 =
```

Valid in Facility.txt

Each turn, the anger levels of every planet in the system are modified by Value1. Only positive values function; negatives are ignored. Only the highest level of this ability on all facilities in the system is used; others are ignored.

```
Ship Training
  Value1 = Per turn increase in ship experience in this sector.
  Value2 = Maximum experience level that can be attained here.
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

Only the highest level of each of the values for this ability on any planet or ship are used in each turn. So, 2 training facilities do not stack on a planet. But, multiple planets or ships in the sector will add their bonuses. So, a planet with moons can have several training facilities functional, thus increasing training rates geometrically.

```
Fleet Training
  Value1 = Per turn increase in fleet experience in this sector.
  Value2 = Maximum experience level that can be attained here.
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

Only the highest level of each of the values for this ability on any planet or ship are used in each turn. So, 2 training facilities do not stack on a planet. But, multiple planets or ships in the sector will add their bonuses. So, a planet with moons can have several training facilities functional, thus increasing training rates geometrically.

```
Modify Reproduction - System
  Value1 = Percentage change in reproduction for entire system.
  Value2 =
```

Valid in Facility.txt

The reproduction rate for all planets owned by this empire in this system is increased by Value1. Only the highest value for this ability is used in the entire system; it does not stack.

```
Change Population - System
  Value1 = Population in M that will be added each turn for entire
system.
  Value2 =
```

Valid in Facility.txt

At the beginning of each turn after normal reproduction has occurred, the populations of all planets owned by this empire in this system is increased by Value1.

```
Plague Prevention - System
  Value1 = Level of the plague that will be prevented in this system.
  Value2 =
```

Valid in Facility.txt

At the beginning of each turn, all plagues of level Value1 or lower that affect a planet owned by this empire in this system will be cured and have no effects.

```
Resource Conversion
  Value1 = Percentage loss of resources converted from one type to
another.
  Value2 =
```

Valid in Facility.txt

This ability allows the Convert Resources ability to be used. The amount of resources received after conversion is equal to 100% - Value1 %

```
Resource Reclamation
  Value1 = Percentage of value of item scrapped in sector returned as
resources.
  Value2 =
```

Valid in Facility.txt

Value1 will supercede the value for resources returned when scrapping a ship that is in Settings.txt.

```
Close Warp Point
  Value1 =
  Value2 =
```

Valid in Components.txt

Allows a ship to use the Close Warp Point order. Can not be used on Bases.

```
Destroy Star
  Value1 =
  Value2 =
```

Valid in Components.txt

Allows a ship to use the Destroy Star order. Can not be used on Bases.

```
Create Star
  Value1 =
```

```
  Value2 =
```

Valid in Components.txt

Allows a ship to use the Create Star order. Can not be used on Bases.

```
Destroy Storm
  Value1 =
  Value2 =
```

Valid in Components.txt

Allows a ship to use the Destroy Storm order, for destroying sector-sized storms. Can not be used on Bases.

```
Create Storm
  Value1 =
  Value2 =
```

Valid in Components.txt.

Allows a ship to use the Create Storm order, for creating sector-sized storms. Can not be used on Bases.

```
Self-Destruct
  Value1 =
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

When the ship/base is successfully boarded, it and the attacking ship will be destroyed. Also, it allows the ship/base to use the Self-Destruct option in the Scrap window.

```
Colonize Planet - Rock
  Value1 =
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

Allows the ship to be used to colonize rock planets. The ship is consumed in the action.

```
Colonize Planet - Ice
  Value1 =
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

Allows the ship to be used to colonize ice planets. The ship is consumed in the action.

```
Colonize Planet - Gas
  Value1 =
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

Allows the ship to be used to colonize gas giant planets. The ship is consumed in the action.

```
Point-Defense
  Value1 =
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

This ability does absolutely nothing. In the stock game, it serves as an AI Tag for the AI to be able to add Point - Defense Cannons to their ship designs as a Misc Ability, so as not to use up a Primary or Secondary weapon slot.

```
Armor
  Value1 =
  Value2 =
```

Valid in Components.txt

All components with the Armor ability must be destroyed before any non-armor skipping damage can damage "internal" components (i.e.: those that do not have the Armor ability). Armor-skipping weapons will not damage these components until all that is left on the ship is Armor.

```
Launch/Recover Satellites
  Value1 = Amount of satellites that can be launched per combat turn.
  Value2 = Amount of satellites that can be launched per game turn.
```

Valid in Components.txt, VehicleSize.txt

Self-explanatory.

```
Remote Resource Generation - Minerals
  Value1 = The number of minerals per turn which can be generated from
a remote source.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

When the ship is in a sector with planets and/or asteroids, it produce a base of Value1 minerals per turn, modified by the planet's value, per planet and asteroid in the sector.

```
Remote Resource Generation - Organics
  Value1 = The number of minerals per turn which can be generated from
a remote source.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

When the ship is in a sector with planets and/or asteroids, it produce a base of Value1 organics per turn, modified by the planet's value, per planet and asteroid in the sector.

```
Remote Resource Generation - Radioactives
  Value1 = The number of minerals per turn which can be generated from
a remote source.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

When the ship is in a sector with planets and/or asteroids, it produce a base of Value1 radioactives per turn, modified by the planet's value, per planet and asteroid in the sector.

```
Armor Regeneration
```

```
  Value1 = The amount of structure points regenerated per combat turn.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

Having this ability with Value1 at least 1 allows the component to regenerate in combat. All of the regeneration abilities for all components with this ability is added up, and this many hit points of damage to such components can be repaired each combat turn. After combat, all components with this ability will be repaired.

```
Shield Generation From Damage
  Value1 = Amount of damage pts converted to shield pts per hit.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

When any weapon does damage to the ship itself (past the shields), Value1 is added to the shield levels for the ship. The most points that can be added to the shields in this manner is how much damage was done to the ship (not to the shields in the first place). Any extra shield points past the maximum from the ship's shield generating components are lost. The weapon does not have to hit the component with this ability on it to trigger the ability. A weapon with either the Skips Armor or Skips Shields And Armor damage type will never trigger this ability.

```
System - Movement Towards Center
  Value1 = # of squares ships are moved towards the center of a system
per turn.
  Value2 =
```

Valid in SystemTypes.txt

The presence of this ability causes all ships to be moved Value1 squares towards the center of the system. In Sequential movement games, this occurs between turns. In Simultaneous movement games, it occurs after normal ship movement.

```
System - Movement Random
  Value1 = # of squares ships are randomly in a system per turn.
  Value2 =
```

Valid in SystemTypes.txt

The presence of this ability causes all ships to be moved Value1 squares towards a random sector in the system. Note that if the target sector is close enough, the ships will not be moved

the full Value1 distance. All ships owned by an empire in a sector will be moved in the same direction, irregardless of fleeting.

```
System - Destructive Center
  Value1 = Amount of damage caused to ships in the center of the system. (Normal Damage)
  Value2 =
```

Valid in SystemTypes.txt

Value1 damage is done each turn that a ship is in the central sector (6,6) of the system at the beginning of the turn.

```
Destroy Nebulae
  Value1 =
  Value2 =
```

Valid in Components.txt

Allows the ship to use the Destroy Nebulae order. Can not be used on Bases.

```
Create Nebulae
  Value1 =
  Value2 =
```

Valid in Components.txt

Allows the ship to use the Create Nebulae order. Can not be used on Bases.

```
Destroy Black Hole
  Value1 =
  Value2 =
```

Valid in Components.txt

Allows the ship to use the Destroy Black Hole order. Can not be used on Bases.

```
Create Black Hole
  Value1 =
  Value2 =
```

Valid in Components.txt

Allows the ship to use the Create Black Hole order. Can not be used on Bases.

```
Stop Planet Destroyer
  Value1 =
  Value2 =
```

Valid in Facility.txt

Prevents any Planet Destroying devices from functioning against the planet the facility is on.

```
Stop Star Destroyer
  Value1 =
  Value2 =
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

Prevents any Star Destroying devices from functioning within the system.

```
Stop Nebulae Creator
  Value1 =
  Value2 =
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

Prevents any Nebulae Creating devices from functioning within the system.

```
Stop Black Hole Creator
  Value1 =
  Value2 =
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

Prevents any Black Hole Creating devices from functioning within the system.

```
Stop Open Warp Point
  Value1 =
  Value2 =
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

Prevents any Warp Point Opening devices from functioning within the system.

```
Stop Close Warp Point
  Value1 =
  Value2 =
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

Prevents any Warp Point Closing devices from functioning within the system.

```
Component Destroyed On Use
  Value1 =
  Value2 =
```

Valid in Components.txt

This causes the component to be destroyed when it is used. It is only functional with abilities that require pressing a button in the main buttons panel on the strategic map. e.g.: Stellar Manipulation, Emergency Movement, etc. Will not work with weapons, cloaking devices, etc.

```
Ancient Ruins
  Value1 = Number of techs areas (random) received when this planet is
colonized.
  Value2 =
```

Valid in StellarAbilityTypes.txt

Causes the planet to give Value1 tech levels to the empire that colonizes it. The techs are determined randomly when the planet is colonized.

```
Ancient Ruins Unique
  Value1 = The unique tech area identifier that is found when the
planet is colonized.
  Value2 =
```

Valid in StellarAbilityTypes.txt

Seeds a planet with a specific Unique Tech. When an empire colonizes the planet, they get level 1 in the tech area corresponding to the Value1 Unique Area (in TechArea.txt).

```
Combat Best Experience
  Value1 =
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

The ship will get the best experience bonus from all ships owned by the empire in a battle that have this ability on them.

```
Combat Movement
  Value1 = Amount of movement added during combat.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

Gives Value1 bonus movement points in combat. Only the best of these abilities on a vehicle is functional. This ability can be used on bases to give them combat movement points.

This ability does not contribute to engine supply usage. Only Standard Ship Movement does.

```
Solar Supply Generation
  Value1 = Amount of supplies generated per star.
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

Each turn, Value1 x number of stars in the system supplies are added to the supply bays of the ship.

```
Extra Movement Generation
  Value1 = Amount of movement generated by a type of component.
  Value2 = Unique identifier for this type
```

Valid in Components.txt, VehicleSize.txt

This provides a bonus amount of movement points to the vehicle. This bonus is independent of the amount of this component placed on the vehicle, as well as the Engines Per Move value of the vehicle. For this ability to function, the vehicle has to have at least 1 movement point generated via the Standard Ship Movement ability for this ability to function (after dividing by the Engines Per Move setting for the ship hull).

Multiple componets (and/or hull abilities) with this ability will stack if and only if they have different Value2 values. If they have the same Value2 value, they will not stack. Only the best Value1 ability will be used from each group of components with this ability, where the groups are based on the Value2 value.

This ability does not contribute to engine supply usage. Only Standard Ship Movement does.

```
Planet - Change Atmosphere
  Value1 = Number of turns until atmosphere is changed.
  Value2 = (It changes to the atmosphere breathable by the majority of the population.)
```

Valid in Facility.txt

After the facility has been in existence for Value1 turns, the atmosphere gets converted to that breathed by the race with the most population on the planet each turn. If there is a tie, no conversion is done.

```
Weapons Always Hit
  Value1 =
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

As long as the component is intact, all direct fire weapons fired from this ship will always hit their targets.

```
Create Constructed Planet
  Value1 = The Special Ability ID in PlanetSize.txt to construct.
  Value2 =
```

Valid in Components.txt

This ability allows a ship to construct a planet from a star. The planet it can construct is related to the Special Ability ID in PlanetSize.txt by the Value1. This ability should be used in conjunction with Constructed Planet Requirements, or else there will be no requirements to construct planets.

```
Constructed Planet Requirements
  Value1 = The custom group of the component needed to be present.
  Value2 = The kT of component required to be present.
```

Valid in Components.txt

This ability must be used in conjunction with the Create Constructed Planet ability. Value1 is the value that determines what type of components will satisfy the requirement, based off of the Custom Group lines in Components.txt. Value2 is how many kilotons of these components are needed. If Value2 is 10000 and the components are 2000 kT, then you will need 5 of those components present in the sector to be able to construct the planet.

```
Modified Maintenance Cost
  Value1 = Percentage of normal maintenance (10% = 110% of normal,
-10% = 90% of normal)
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

This modifies the maintenance paid of the ship, based on the base maintenance rate of the ship after taking racial modifiers into account. So, if a race has 110 maintenance aptitude, then a ship costing 1000 minerals will cost 150 to maintain each turn. If that ship has the Modified Maintenance Cost ability at 25%, then the ship costs 25% fewer maintenance points, and so only 113 minerals are paid per turn to maintain it. This ability can get the cost to maintain a ship below the 5% minimum imposed from racial modifiers.

```
Ship Training - System
  Value1 = Per turn increase in ship experience in this system.
  Value2 = Maximum experience level that can be attained here.
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

Value1 experience will be added to every ship or base in the system, as long as the ship or base does not have Value2 or more experience already.

```
Fleet Training - System
  Value1 = Per turn increase in fleet experience in this system.
  Value2 = Maximum experience level that can be attained here.
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

Value1 experience will be added to every fleet in the system, as long as the fleet does not have Value2 or more experience already.

```
Long Range Scanner - System
  Value1 =
  Value2 =
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

This ability allows long range scans to be performed on any ship in the same system, regardless of how far away the ship is.

```
Solar Resource Generation - Minerals
  Value1 = Amount of minerals generated per turn per star in the system.
  Value2 =
```

Valid in Facility.txt

Value1 minerals are produced per star in the system per turn. This amount is not affected by planet value, happiness, population, racial characteristics, or other modifiers.

```
Solar Resource Generation - Organics
  Value1 = Amount of organics generated per turn per star in the system.
  Value2 =
```

Valid in Facility.txt

Value1 organics are produced per star in the system per turn. This amount is not affected by planet value, happiness, population, racial characteristics, or other modifiers.

```
Solar Resource Generation - Radioactives
```

```
  Value1 = Amount of radioactives generated per turn per star in the
system.
  Value2 =
```

Valid in Facility.txt

Value1 radioactives are produced per star in the system per turn. This amount is not affected by planet value, happiness, population, racial characteristics, or other modifiers.

```
Reduced Maintenance Cost - System
  Value1 = Percentage of normal maintenance for all ships in system
(10 = 90% of normal, -10 = 110% or normal)
  Value2 =
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

The maintenance paid on all ships and bases in the system owned by the empire is lowered by Value1 percent. This ability can get the cost to maintain a ship below the 5% minimum imposed from racial modifiers.

```
Shield Modifier - System
  Value1 = Amount added to maximum shields of players ships during
combat.
  Value2 =
```

Valid in Components.txt, Facility.txt, VehicleSize.txt

This ability does not differentiate between phased and non-phased shielding. Ships with phased shields still have phased shields after this ability is applied.

```
Combat To Hit Offense Minus
  Value1 = Percent negative modifier for combat offense.
          (Decreases the chance that this ship will hit another ship).
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

This ability subtracts Value1 % from the chance of this vehicle hitting its target with direct-fire weapons. The modifier subtracts linearly, so 50% - 30% = 20%

```
Combat To Hit Defense Minus
  Value1 = Percent negative modifier for combat defense.
          (Increases the chance that this ship will be hit by another
ship).
  Value2 =
```

Valid in Components.txt, VehicleSize.txt

This ability adds Value1 % to the chance of enemy direct-fire weapons hitting this vessel. The modifier adds linearly, so 50% + 30% = 80%

```
AI Tag 01
  Value1 =
  Value2 =
```

Valid in Components.txt, Facility.txt

These 20 abilities (up to AI Tag 20) do nothing. They serve as extra tags that modders can add to a component or facility for getting the AI to use a specific item instead of other possibilities. One example would be adding an Advanced Combat Sensor component, which adds a to hit bonus and stacks with normal combat sensors. Ideally, warships would use both, to get the maximum level of to hit bonus. However, the AI will only use on or the other. So, you can add an AI Tag to the one that gives less to hit bonus, and add calls for it in AI_Design_Creation.txt. Now, the AI will be able to add both a Combat Sensor and an Advanced Combat Sensor, just as a human would be able to.

```
Generate Points Minerals
  Value1 = The number of minerals per turn which can be generated.
  Value2 = (This ability can be used by a component or a facility. The
points
           are not deducted from any planet.)
```

Valid in Components.txt, Facilities.txt, StellarAbilityTypes.txt, VehicleSize.txt

This ability generates Value1 Minerals every turn. The amount is not modified by any modifiers, whether they are racial characteristics, facilities with "Resource Gen Modifier Planet - Minerals" type abilities, or any others. The points generated by this ability do not show up in the empire status window, nor on the various reports of the ship/planet/etc. that the ability is located on. The points are added to the stored resource totals correctly every turn; they are just not tallied up in the display properly.

```
Generate Points Organics
```

```
  Value1 = The number of organics per turn which can be generated.
  Value2 = (This ability can be used by a component or a facility. The
points
           are not deducted from any planet.)
```

Valid in Components.txt, Facilities.txt, StellarAbilityTypes.txt, VehicleSize.txt

This ability generates Value1 Organics every turn. The amount is not modified by any modifiers, whether they are racial characteristics, facilities with "Resource Gen Modifier Planet - Organics" type abilities, or any others. The points generated by this ability do not show up in the empire status window, nor on the various reports of the ship/planet/etc. that the ability is located on. The points are added to the stored resource totals correctly every turn; they are just not tallied up in the display properly.

```
Generate Points Radioactives
  Value1 = The number of radioactives per turn which can be generated.
  Value2 = (This ability can be used by a component or a facility. The
points
           are not deducted from any planet.)
```

Valid in Components.txt, Facilities.txt, StellarAbilityTypes.txt, VehicleSize.txt

This ability generates Value1 Radioactives every turn. The amount is not modified by any modifiers, whether they are racial characteristics, facilities with "Resource Gen Modifier Planet - Radioactives" type abilities, or any others. The points generated by this ability do not show up in the empire status window, nor on the various reports of the ship/planet/etc. that the ability is located on. The points are added to the stored resource totals correctly every turn; they are just not tallied up in the display properly.

```
Generate Points Research
  Value1 = The number of research points per turn which can be
generated.
  Value2 = (This ability can be used by a component or a facility. The
points
           are not deducted from any planet.)
```

Valid in Components.txt, Facilities.txt, StellarAbilityTypes.txt, VehicleSize.txt

This ability generates Value1 research points every turn. The amount is not modified by any modifiers, whether they are racial characteristics, facilities with "Planet Point Generation Modifier - Research" type abilities, or any others. The points generated by this ability show up in the research window correctly, but not on the various reports of the ship/planet/etc. that the ability is located on.

```
Generate Points Intelligence
  Value1 = The number of intelligence points per turn which can be
generated.
  Value2 = (This ability can be used by a component or a facility. The
points
           are not deducted from any planet.)
```

Valid in Components.txt, Facilities.txt, StellarAbilityTypes.txt, VehicleSize.txt

This ability generates Value1 intelligence points every turn. The amount is not modified by any modifiers, whether they are racial characteristics, facilities with "Planet Point Generation Modifier - Intelligence" type abilities, or any others. The points generated by this ability show up in the intelligence window correctly, but not on the various reports of the ship/planet/etc. that the ability is located on.

Return to Table of Contents

Chapter 17: Advanced Tips and Tricks

Introduction and Contents

This chapter is about some advanced modding tricks. It is recommended that you have a good understanding of how modding Space Empires IV works before you read through them.

Quasi-Newtonian Propulsion

Background:
==========
Newtonian physics would have that the acceleration of a ship be dependent on the thrust of its engines divided by its mass. (F=ma) Unmodded SE4 uses a system where Speed is equal to the thrust of the engines, no matter what the mass of the ship.
Quasi-Newtonian Propulsion is an attempt to improve on the realism of SE4's propulsion system, while still being possible using only the moddable data files. Since SE4 does not model acceleration in its movement system, the QNP compromise is that speed should be dependent on thrust divided by mass. (F=mv)

SE4 limitations:
==========
There are a few limitations in SE4 that you must be aware of when implementing a QNP scheme.

1) Acceleration is not modeled, so F=mv is the closest to Newtonian physics that can be achieved in SE4.
2) There is a maximum of 255 "standard movement" points allowed on a design without causing a Range Check Error. This restricts the scale of your QNP scheme, and may limit the maximum speed of larger ships.
3) Only integer values for "standard movement" points are allowed.
4) Ship speeds above 30 movement points will not be useful in simultaneous-turn games. Ships of over 30 speed will only move once per "day", and will cover at most 30 sectors.

Calculating Speeds in SE4:
==========
To calculate the final speed of a ship in SE4, the following method is used:
1) Add up all "Standard Movement" points on a ship.
2) Divide by the "Engines Per Move" setting on the vehicle, rounding DOWN.
3) If the speed of the vessel is at least one, add any "Bonus Movement" points, and add 1 speed if the race has the "Propulsion experts" trait chosen.

Designing a QNP scale:
==========
Any QNP model requires a scale. This only fixes the numbers that will be used internally by the mod, and does not determine how much space is required on a ship to move at a specific speed.

NOTE: Since ships and fighters use completely separate engine components, the mod can and should have a separate scale for both types of vehicle.

If the scale is very large, then bigger ships will be able to be packed with engines without causing Range Check Errors. Unfortunately, a large scale may cause inaccuracies due to rounding errors. Given the default ship sizes in SE4, a value of 1 "standard move" per 50kt of ship providing 1 Speed is the best choice. If the escort were changed to a size of 100kt or removed entirely, then 1 "standard move" per 100kt of ship providing 1 speed would be an option. For fighters, a value of 1 "standard move" per 5kt of fighter providing 1 speed would likely be the best choice. In any case, the value you choose should divide evenly into all hull sizes (or as many as possible).

Implementing the QNP system:
==========
Once you have chosen a scale for your QNP system, you can start editing the mod files.
Open VehicleSize.txt, and for each ship:
1) Find the Hull size in kT.
2) Divide the hull size by your scale, determined above.
3) Set the "Engines per Move" setting to the value calculated in step 2
4) Set the "Requirement Max Engines" to 255.
Go back through the file, and repeat the process for all fighters, using the fighter scale chosen above.

Balancing the QNP system:
==========
If you were to start a game you should quickly find that larger ships do indeed require many more engines to reach a particular speed than smaller ships. However, you will also notice that it takes an incredibly large number of unmodded engines to achieve even speed 1.
It is time to mod the engine components. What you will have to do is increase the number of "Standard movement" points each engine component provides. Too much, and your ships will zoom through the galaxy at 30+ sectors per turn. Too few, and your ships will be giant clusters of

engines that can barely crawl through space.

A good choice for a scale of 50 is 3 Standard movement on an Ion engine. The bonus movement points provided by the higher-tech engines are clearly not Q-N. Instead of providing a bonus movement, simply have the engine provide more standard movement! If an Ion engine provides 3 standard movement, you could have a Contraterrene drive provide 4, a Jacketed Photon drive 5, and Quantum engines 6. Thus an escort (3 engines per move) would need 4 Ion engines to move at speed 4, but only 3 CT drives to move the same speed, and only 2 Quantum drives to move speed 4. Alternatively, you could reduce the size of the higher-tech engines, while they provide the same amount of thrust. There are many ways to balance the speed of the engines at this point, just be creative.

Once you have modded the engines, be sure to check that the maximum speeds are all reasonable throughout the range of hull sizes. Be sure that escorts cannot travel beyond a speed of 30, and that the largest ships can actually move.
Note: the fastest possible hull size is not actually the escort! While the escort must spend almost 20% of its hull space on bridge/life support/crew quarters, a Light Cruiser spends only 8%. Under a QNP system, and given equal technology, a larger ship can go faster if all of its free space is devoted to engines.

Some other components that you will have to alter will be:
- Solar sails
- Afterburners
- Higher-tech fighter engines.

QNP side-effects:
==========
When implementing a QNP system some common side effects will show up.
- Larger ships will burn vast amounts of supplies moving from one sector to another. Since they require more engines, they use up proportionally more supplies. This means that it will take exactly the same amount of supplies to move Two 400kt ships from point A to point B as it will to move One 800kt ship from A to B at the same speed.
- Smaller ships remain useful late in the game. The fact that large ships must spend about the same fraction of their space on engines as the little ships means that given fleets of equal mass, the fleet of small ships will carry roughly the same tonnage of ordnance as the fleet of large ships. The large ships will have mounts and be harder to kill individually, but the smaller ships will have a defense bonus due to size.
- The final speed of a ship will be totally dependent on the designer. Ship designs will be more diverse, with very fast scout ships composed only of engines and supply tanks, and slower warships that carry extra firepower or shields. Each race may have a different opinion of the optimum balance between engines, shields and weapons.

"Quasi-Newtonian Propulsion" by Nick Dumas (Suicide Junkie), August 18, 2002.

Leaky Armor

Background:
==========
Have you ever wondered why all armor on your ship, no matter which facing it is on, must be completely vaporized before any damage can be done to internal components (other than from

armor skipping weapons)? The solution to this conundrum is leaky armor. Leaky armor is armor in the sense that it protects your ship from physical harm. However, it is not armor in the sense that it does not have the Armor ability. It will protect from some weapon hits, and not protect against others. Some hits will leak through the armor layer to damage internal components. Thus, there is no magic wall to be penetrated as with normal armor. Leaky type armor first appeared in the Pirates & Nomads mod, in the form of Bucky-Tube Gel Plating.

How it works mechanically:
==========
No component with the Armor ability will protect you from weapons with the Skips Armor or Skips Shields and Armor damage types. Leaky armors treat armor-skipping weapons as normal weapons, as there is no armor layer to pierce. Components with the Armor ability are only damaged by armor skipping weapons once there are no other components left intact on the ship that do not have the armor ability.

When a weapon hits a ship, it will first face the shield layer. If there are no shield generators, or no shield points, the damage goes to the components of the ship. If there are any components with Armor ability, they will be hit first. But, if the weapon was one with armor-skipping damage, such as Shard Cannons, then the damage will completely bypass the Armor layer and damage "internals," where internals are any components that do not have the Armor ability. Only the Armor ability makes a component act like "armor."

If a ship has "leaky armor" components, they are not actually "armor" because they do not have the Armor ability. They are treated exactly like other internals. The reason that we use the term "leaky armor" is because having some beefed up components (lots of hit points) without the Armor ability makes them act similar to armor, except that some shots will not hit them but hit other internals instead. So instead of complete absorption by the Armor, you have partial absorption by the leaky armor.

A note on abilities:
==========
Armor-skipping damage has one role, to bypass the Armor layer. Or, in other words, those components with the Armor ability. However, testing has shown a few other side effects. The abilities of Shields From Damage and Emissive Armor do not get triggered by weapon shots with the armor-skipping damage type. Both of these abilities will function when any component gets hit, even if that component does not have that ability. This is why combining stock Armor and Emissive Armor works. Any shot from a regular weapon that hits any armor component will be emissed by so many damage points. The same occurs with Crystalline Armor, which has the shields from damage ability. If you assign either of these abilities to an internal component, then that ability will be triggered when any component on the ship is damaged. Whether it is internal or armor is irrelevant. But, their effects do not get triggered from weapon damage of the armor-skipping type. So, if you mod in an internal component that has the Shields From Damage ability (such as leaky shields as in some mods), no shields will be added from the damage occurred by an armor-skipping weapon (such as Shard Cannons).

Implementation:
==========
To implement leaky armor, create a component that has no abilities. Give it more structure (hit points) than the average internal ship component has. This way, it is more likely to be hit first than other internal components. If the average component has 10 hit points, then leaky armor with 15 hit points will have a good chance of being hit first. But, a lot of shots will still hit other internals.

Since the only purpose of leaky armor is to absorb damage, it should have a much higher structure to space taken ratio than other components. If it does not, then those other components might end up as being better leaky armor than your new armor! The exact values are up to you, but I would recommend as a bare minimum that all leaky armor components have at least 2 times the structure to space taken ratio as most internals do. In stock SE4, most internals have a ratio of either 1:1 or 2:1 (engines being those with 2:1). So, 4:1 is a good minimal ratio for your leaky armor

Complexity:
==========
You can implement a system where there is just one type of leaky armor. Or, you can implement a system with multiple types, to create more strategic diversity. Let's take a system with two types of leaky armor as an example. Light Armor is 1 kiloton in size and has, say, 20 structure (hit points). Heavy Armor is 10 kilotons in size and has 100 structure.

In this system, the Light Armor has twice as many hit points per kiloton of space taken as the Heavy Armor. This means that it should create ships much stronger than Heavy Armor can, right? Well, not quite. Even though the Light Armored ships can take a lot more damage, you will find that the heavy armored ships tend to win more often. This is because weapon hits are more likely to strike a component of Heavy Armor than other internal components with the HA ship than they are to strike a component of Light Armor than an internal on a LA ship. The large amount of hit points that the Heavy Armor has individually gives it a greater chance to be struck. So while the LA ships can take more damage overall, they tend to be more leaky, so vital internals (engines, weapons) tend to be struck more quickly than they do on the HA ships. The Heavy Armor has more hit points than any (unmounted) weapon, after all. Of course, it turns out that the best design is a combination of LA and HA.

Using this example, you can create complex armor systems. More complexity (to a point) means that there are more strategic options when designing and employing your ships. This in turn leads to a greater depth of game play. Of course, you should avoid going overboard. 12 different types of armor (not levels, but different types entirely) would probably be too many.

"Leaky Armor" by Nolan Kelly (Imperator Fyron), February 23, 2004.

Leaky Shields

Background:
==========
Just like armor, shields in stock SE4 are a magic wall that blocks all damage until depleted. Just like armor, however, you can mod the shields to be "Leaky". Leaky shields will typically absorb an average of 50% of the damage from normal weapons, although this can vary from 0% to 100% depending on the implementation details (See below).

Leaky shields provide a nice alternative to the all-or-nothing stock shield system. Combined with leaky armor, you will get ships that will occasionally take critical damage right from the first hit if they are particularly unlucky. Backup internals become very important, and combat will tend to involve many ships doing the best they can with a handful of damaged components each.

Mechanics:
==========

The "shield generation from damage" ability is the core of the leaky shields effect. Each time a non-armor-skipping weapon hits the hull of a ship, all of the SGFD ability points are used to boost the shield points of the damaged ship. SGFD will add a number of shield points equal to the lowest of:
(A) the hull damage sustained from this hit (shield damage does not count)
-OR-
(B) the total SGFD ability points from all undamaged components on the damaged ship.

Note: The hull damage is NOT negated, and the ship will be left with some shields remaining AND some components destroyed or partially damaged. However, any partial damage will be added to the next hit and may trigger the SGFD ability again.

Implementation:
==========
Shield generators will be altered to have two abilities instead of just one. They will need both "Shield Generation From Damage" AND "Shield Generation". You may optionally replace the latter with "Phased Shield Generation" if you want phased weapons to participate in the leaky shields effect instead of skipping completely.

The value1 amounts of both abilities should typically be set to the same number. The shield generation amount must be equal or greater than the SGFD amount. If you set the shield generation amount higher than the SGFD amount, you will introduce a small amount of stock-like "solid shield" effect for the first few hits. Once the excess shield points are depleted, the "leaky shield" effect will take over from the "solid shield" effect

The actual value amounts of the leaky shield abilities will depend on many factors, including the size of the shield component, the weapons and mounts available to use against it, the size of the hull, and the armor system. Values of 1-3 ability points per kT of size is a good starting value, but will certainly need play testing to find appropriate values for your mod.

Be sure to adjust the descriptions to match the new effects of your shield generators!

Qualitative Effects:
==========
There are a number of different effects that you will see when using Leaky Shields. Unless otherwise noted, these effects are seen when averaged over multiple hits from the same weapon

1) If a ship takes individual hits with damage *greater* than twice the number of leaky shield points, then the leaky shields will act like emissive armor ability. They will reduce the damage sustained from each hit by an amount equal to the count of leaky shield points. This emissive style protection will be multiplied or divided by 2 or 4 if the weapon uses the quad/double/half/quarter to shields damage types.

2) If a ship takes individual hits with damage *less* than twice the number of leaky shield points, then the following applies:

Damage Type	**Average Damage to Hull/Armor**
Quarter To Shields	20% of normal
Half To Shields	33% of normal

Normal	50% of normal
Phased	50% or 100% of normal
Double To Shields	67% of normal
Quad To Shields	80% of normal
Skips Shields/Armor	100% of normal
Only "X"	100% of normal

3) If the damage applied to the hull is not sufficient to destroy components, then the partial damage left over will be applied to the next hit. The partial damage must pass through the shields again, and the shields will get a second chance to block about half of it (for normal damage weapons, see chart for others). In effect, the first shot has just been blocked 75% by the shields instead of 50%. With sufficiently high-hitpoint armor/internals (2x the weapon's damage per hit) and enough leaky shield generation (2x the weapon's damage per hit again), the shield will be able to block 100%!

If you are using leaky armor, it is fairly easy to have enough shield/armor points to cause this, but since the armor is leaky, the weapons will occasionally hit a shield generator instead of the leaky armor, and the system will eventually collapse as the shields are destroyed.

"Leaky Shields" by Nick Dumas (Suicide Junkie), February 26, 2005.

Maintenance Vs. Construction Cost Tradeoffs

In addition to Quasi-Newtonian Propulsion, there are other ways to force players to make tradeoffs between the advantages of small and large ships: maintenance Vs. construction costs. In other words, small ships are cheap to build but expensive to maintain (making them good for "on the fly" construction during times of war), while big ships are expensive to build but cheap (for their size) to maintain (making them good to build up during peacetime as a less micromanagement-intensive alternative to mothballing). There are basically two ways of going about this: one as implemented in Ed Kolis' Economies of Scale mod, and the other as implemented in Suicide Junkie's as-yet-unnamed "new SJ mod".

Ed's method is to assign some arbitrary "average component cost per kiloton" value, multiply by the tonnage of each hull, then subtract that value from the desired construction cost of each hull (so if you want a squares progression you might have ES = 1000, FG = 4000, DS = 9000, LC = 16000, etc.), to get the construction cost for each hull. Then, take the desired maintenance amount for each hull (you might use an approximately square root progression of ES = 100, FG = 140, DS = 170, LC = 200) and assign a maintenance modifier as appropriate to get the hulls' desired maintenance costs to equal the appropriate proportion of their respective construction costs. (So for our example if we used a base maintenance cost in Settings.txt of 10% we'd have maintenance reductions of ES = 0%, FG = 65%, DS = 81%, LC = 88%.) To find the maintenance reductions, take BM = base maintenance (base maintenance cost as a percentage * construction cost) and DM = desired maintenance; then the maintenance reduction is (BM - DM) / BM * 100%.

SJ's method is slightly different; instead of increasing the cost of the hulls, he increases the cost of the components by use of mounts. Each component is given a cost 100 times what it would cost to be installed on the smallest ship, and the mounts scale the cost down to something reasonable, while keeping the tonnage, etc. the same. Then, to scale the maintenance costs down, he gives the hulls inherent maintenance reduction much like Ed does.

So which method is better? Well, it depends on your situation. Does your mod have a lot of mounts? Then SJ's method might be a bit troublesome to implement, because you'll need a copy of every mount for every hull size you it can be used on, just with a different cost scale factor (e.g. LC Large Mount, CR Large Mount, BC Large Mount, etc.) However, Ed's method does have the problem of components with unusual costs - since everything is calculated based on average component costs, what happens when you have a dreadnought filled with repair bays, or a frigate with a star destroyer component? Your numbers get thrown off and you either get ripped off (in the case of the dreadnought) or an incredible maintenance bargain (in the case of the frigate). But that's not entirely a bad thing - it adds more thought to decisions like "do I beef up my expensive stellar manipulation ships or go lean & mean with them to save money. So it's up to you - do you want those stellar manipulation components to be cheaper to build on small hulls (SJ's version) or cheaper to maintain (Ed's)? ;-)

"Maintenance Vs. Construction Cost Tradeoffs" by Edward Kolis (Ekolis), February 23, 2004.

Space Yard Expansions

Background:
==========
In Space Empires IV, there is a hard-coded limit of only being able to construct one Space Yard facility per planet. This is done to prevent stacking 30 space yard facilities on a planet and get extremely expensive ships (such as black hole creators) built in a single turn. However, many games have the ability to have graduated construction rates based on the number of facilities present. Each facility adds a small amount of construction (such as 200 points), rather than a huge chunk as with a stock SEIV space yard (2000 to 3000). This can be accomplished in Space Empires IV, after a fashion.

How it works mechanically:
==========
While you can not build a facility with a Space Yard ability if there is already one on the planet, this restriction does not apply to facility upgrades. It is entirely possible to upgrade a non-SY facility to a SY facility and bypass the restriction. The rates of all space yards on the planet stack together.

Implementation:
==========
Space Yard Expansions are very simple to implement. There are several methods, depending on preference.

The easiest, which I strongly discourage you from using, is to add a facility with no abilities and the same family number as an existing Space Yard facility. It must be "less advanced," meaning that it should have a lower roman numeral (such as 0). After building this facility, you can then upgrade it to a Space Yard. Keep in mind that the cost of an upgrade (with stock upgrade cost settings) is one half that of building a facility directly, so this method could very easily allow you

to build space yards far more quickly than normal, depending on how much the base facility costs.

A better method is to have normal Space Yards and a new type of facility, the Space Yard Expansion. A Space Yard Expansion should provide a small, incremental amount of construction, such as one tenth of that of a normal space yard. Exact values are up to the modder's discretion, of course. To implement this, you need to add two new facilities, with a new, unused family number. Do not make these have the same family number as normal space yards. The components can be called "Space Yard Expansion Project" and "Space Yard Expansion." The Space Yard Expansion Project will have no abilities. It is meant as the facility you build, then upgrade to the Space Yard Expansion facility. Give the Space Yard Expansion Project a roman numeral value of 0. The Space Yard Expansion facilities should have roman numerals of 1 or greater. Make sure to place the Space Yard Expansion Project after the Space Yard Expansion facilities in Facilities.txt. This will cause SEIV to display them as the only facility in the family when obsolete facilities are hidden. You never want to build the Space Yard Expansion directly, so this makes it much easier to use the system (no need to show obsolete facilities to build the Project facility first). In this sort of system, you would first build a normal space yard, then start with the expansions.

Another method is similar to the second, except that you have 3 different families of Space Yard Expansions, one for each resource. It is set up in much the same way. Make sure to _never_ set a space yard ability value to 0. If the space yard rate of a planet is 0 (and it is not rioting), an unlimited number of resources will be put into construction projects, allowing 1 turn builds of anything (assuming all resource rates are 0).

A final method would be to have no normal space yards. All space yards have small, incremental build values, much like Space Yard Expansion outlined above. You would build them in the same manner as a Space Yard Expansion, though it might be better to call them "Industrial Facilities" or something of that nature, since they are not really expansions anymore. This method changes construction rate to be more gradual. You might wish to lower the base construciton rate without any facilities if you choose to implement this method. Otherwise, you could quite easily see construction rates drop from 2000 to 200 after building the first facility (assuming 200 is the value you chose).

"Space Yard Expansions" by Nolan Kelly (Imperator Fyron), March 23, 2005.

Return to Table of Contents
