Phobos

The Phobos Contributors

Jul 14, 2021

PROJECT INFO

This page lists general info that should be known about the project.

CHAPTER

ONE

BUILD TYPES

There are three main types of Phobos builds:

- *stable builds* - those are numbered like your regular versions (something close to semantic versioning, e.g. version 1.2.3 for example) and ideally should contain no bugs, therefore are safe to use in mods;
- *development builds* - those are the builds which contain functionality that needs to be tested. They are numbered plainly starting from 0 and incrementing the number on each release. Mod authors still can include those versions with their mods if they want latest features, though we can't guarantee lack of bugs;
- *nightly builds* - bleeding edge versions which can include prototypes, proofs of concepts, scrapped features etc., in other words - we can't guarantee anything in those builds and they absolutely should NOT be used in mod releases and should only be used to help with development and testing.

1.1 Disabling development build warning

DISCLAIMER: We understand that everyone wants to try and use the new features as soon as they're released, but we can't do all the testing ourselves, so we only test the functionality on a basic level. We ask everyone who uses the new development build first to **test the new changes in every possible way first before disabling the development build warning** and proceeding to include the build in your mod release. This would allow us to concentrate on implementing the actual features, which is the most complex task. Learn more on testing *here*.

You can hide the warning by specifying the build number after `-b=` as a command line argument (for example, `-b=1` would hide the warning for development build #1 of Phobos).

CHAPTER

TWO

SAVED GAMES FILTERING

Phobos fully supports saving and loading thanks to prototype code from publicly released Ares 0.A source and it implements it's own filtering which shouldn't conflict with Ares save filtering. Save games between different versions are incompatible due to changes to Phobos extension classes which are present in almost every build release. The filtering mechanism, hovewer, doesn't apply to nightly versions - those use latest development build number on which this nightly is based on. While different nightly version saves may be listed, they are most likely incompatible in case there were changes to extension class fields.

CHAPTER

THREE

COMPATIBILITY

While Phobos is standalone, it is designed to be used alongside Ares and CnCNet5 spawner. Adding new features or improving existing ones is done with compatibility with those in mind.

While we would also like to support HAres we can't guarantee compatibility with it due to the separation of it's userbase and developers from international community. We welcome any help on the matter though!

CHAPTER

FOUR

WHAT'S NEW

This page lists the history of changes across stable Phobos releases and also all the stuff that requires modders to change something in their mods to accomodate.

4.1 Migrating

4.1.1 From vanilla

- SHP debris hardcoded shadows now respect `Shadow=no` tag value, and due to it being the default value they wouldn't have hardcoded shadows anymore by default. Override this by specifying `Shadow=yes` for SHP debris.
- Radiation now has owner by default, which means that radiation kills will affect score and radiation field will respect `Affects...` entries. You can override that with `rulesmd.ini->[SOMEWEAPONTYPE]->Rad.NoOwner=yes` entry.

4.1.2 From older Phobos versions

- Key `rulesmd.ini->[SOMETECHNOTYPE]->Deployed.RememberTarget` is deprecated and can be removed now, the bugfix for `DeployToFire` deployers is now always on.

4.1.3 For Map Editor (Final Alert 2)

In `FAData.ini`:

```
[ParamTypes]
47=Structures,28
53=Play BuildUp,10

[ActionsRA2]
125=Build at...,-10,47,53,0,0,0,1,0,0,[LONG DESC]

[ScriptsRA2]   ; NEEDS FA2EXT.DLL (by AlexB) or FA2SP.DLL (by secsome)
71=Timed Area Guard,4,0,1,[LONG DESC]          ; FA2Ext.dll only
71=Timed Area Guard,20,0,1,[LONG DESC]         ; FA2sp.dll only
72=Load Onto Transports,0,0,1,[LONG DESC]
73=Wait until ammo is full,0,0,1,[LONG DESC]
```

4.2 Changelog

4.2.1 0.2.1.1

Phobos fixes:

- Fixed occasional crashes introduced by `Speed=0` stationary vehicles code (by Starkku)

4.2.2 0.2.1

New:

- Setting VehicleType `Speed` to 0 now makes game treat them as stationary (by Starkku)

Vanilla fixes:

- Fixed the bug when after a failed placement the building/defence tab hotkeys won't trigger placement mode again (by Uranusian)
- Fixed the bug when building with `UndeployInto` plays `EVA_NewRallypointEstablished` while undeploying (by secsome)

Phobos fixes:

- Fixed the bug when trigger action `125 Build At...` wasn't actually producing a building when the target cells were occupied (by secsome)

4.2.3 0.2

New:

- Shield logic for TechnoTypes (by Uranusian, secsome, Belonit) with warhead additions (by Starkku)
- Custom Radiation Types (by AlexB, Otamaa, Belonit, Uranusian)
- New ScriptType actions `71 Timed Area Guard`, `72 Load Onto Transports`, `73 Wait until ammo is full` (by FS-21)
- Ore drills now have customizable ore type, range, ore growth stage and amount of cells generated (by Kerbiter)
- Basic projectile interception logic (by AutoGavy, ChrisLv_CN, Kerbiter, Erzoid/SukaHati)
- Customizable harvester active/total counter next to credits counter (by Uranusian)
- Select Next Idle Harvester hotkey command (by Kerbiter)
- Dump Object Info hotkey command (by secsome, FS-21)
- Remove Disguise and Remove Mind Control warhead effects (by secsome)
- Custom per-warhead SplashLists (by Uranusian)
- `AnimList.PickRandom` used to randomize `AnimList` with no side effects (by secsome)
- Chance-based critical damage system on warheads (by AutoGavy)
- Optional mind control range limit (by Uranusian)
- Multiple mind controllers can now release units on overload (by Uranusian, secsome)
- Spawns now can be killed on low power and have limited pursuing range (by FS-21)
- Spawns can now have the same exp. level as owner techno (by Uranusian)

- `TurretOffset` now accepts `F,L,H` and `F,L` values instead of just `F` value (by Kerbiter)
- ElectricBolt arc visuals can now be disabled per-arc (by Otamaa)
- Semantic locomotor aliases for modder convenience (by Belonit)
- Ability to specify amount of shots for strafing aircraft and burst simulation (by Starkku)
- Customizeable Teleport/Chrono Locomotor properties per TechnoType (by Otamaa)
- Maximum waypoints amount increased from 702 to 2147483647 (by secsome)
- Customizeable Missing Cameo file (by Uranusian)

Vanilla fixes:

- Map previews with zero size won't crash the game anymore (by Kerbiter, Belonit)
- Tileset 255+ bridge fix (by E1 Elite)
- Fixed fatal errors when `Blowfish.dll` couldn't be registered in the system properly due to missing admin rights (by Belonit)
- Fix to take Burst into account for aircraft weapon shots beyond the first one (by Starkku)
- Fixed the bug when units are already dead but still in map (for sinking, crashing, dying animation, etc.), they could die again (by Uranusian)
- Fixed the bug when cloaked Desolator was unable to fire his deploy weapon (by Otamaa)
- Fixed the bug when `InfiniteMindControl` with `Damage=1` will auto-release the victim to control new one (by Uranusian)
- Fixed the bug that script action `Move to cell` was still using leftover cell calculations from previous games (by secsome)
- Fixed the bug when trigger action `125 Build At...` didn't play buildup anim (by secsome)
- Fixed `DebrisMaximums` (spawned debris type amounts cannot go beyond specified maximums anymore) (by Otamaa)
- Fixes to `DeployFire` logic (`DeployFireWeapon`, `FireOnce`, stop command now work properly) (by Starkku)

Phobos fixes:

- Properly rewritten a fix for mind-controlled vehicles deploying into buildings (by FS-21)
- Properly rewritten `DeployToFire` fix, tag `Deployed.RememberTarget` is deprecated, now always on (by Kerbiter)
- New warheads now work with Ares' `GenericWarhead` superweapon (by Belonit)

4.2.4 0.1.1

- Fixed an occasional crash when selecting units with a selection box

4.2.5 0.1

New:

- Full-color PCX graphics support (by Belonit)
- Support for PCX loading screens of any size (by Belonit)
- Extended sidebar tooltips with descriptions, recharge time and power consumption/generation (by Kerbiter, Belonit)
- Selection priority filtering for box selection (by Kerbiter)
- Shroud, reveal and money transact warheads (by Belonit)
- Custom game icon command line arg (by Belonit)
- Ability to disable black spawn position dots on map preview (by Belonit)
- Ability to specify applicable building owner for building upgrades (by Kerbiter)
- Customizable disk laser radius (by Belonit, Kerbiter)
- Ability to switch to GDI sidebar layout for any side (by Belonit)

Vanilla fixes:

- Deploying mind-controlled TechnoTypes won't make them permanently mind-controlled anymore (unfinished fix by DCoder)
- SHP debris hardcoded shadows now respect `Shadow=no` tag value (by Kerbiter)
- `DeployToFire` vehicles won't lose target on deploy anymore (unfinished fix by DCoder)
- Fixed QWER hotkey tab switching not hiding the displayed tooltip as it should (by Belonit)
- Sidebar tooltips now can go over sidebar bounds (by Belonit)
- Lifted stupidly small limit for tooltip character amount (by Belonit)

CHAPTER

FIVE

CONTRIBUTING

This page describes ways to help or contribute to Phobos and lists the contributing guidelines that are used in the project.

5.1 Guidelines for contributors

5.1.1 Project structure

Assuming you've successfully cloned and built the project before getting here, you should end up with the following project structure:

- `src/` - all the project's source code resides here.
  - `Commands/` - source code for new hotkey commands. Every command is a new class that inherits from `PhobosCommandClass` (defined in `Commands.h`) and is defined in a separate file with a few methods and then registered in `Commands.cpp`.
  - `New/` - source code for new ingame classes.
    - `Type/` - new enumerated types (types that are declared with a list section in an INI, for example, radiation types) implemented in the project. Every enumerated type class inherits `Enumerable<T>` (where `T` is an enum. type class) class that is defined in `Enumerable.h`.
    - `Entity/` - classes that represent ingame entities are located here.
  - `Ext/` - source code for vanilla engine class extensions. Each class extension is kept in a separate folder named after vanilla engine class name and contains the following:
    - `Body.h` and `Body.cpp` contain class and method definitions/declarations and common extension hooks. Each extension class must contain the following to work correctly:
      - `ExtData` - extension data class definition which inherits `Extension<T>` from `Container.h` (where `T` is the class that is being extended), which is the actual class that contains new data for vanilla classes;
      - `ExtContainer` - a definition of a special map class to store and look up `ExtData` instances for base class instances which inherits `Container<T>` from `Container.h` (where `T` is the extension data class);
      - `ExtMap` - a static instance of `ExtContainer` map;
      - constructor, destructor, serialization, deserialization and (for appropriate classes) INI reading hooks.
    - `Hooks.cpp` and `Hooks.*.cpp` contain non-common hooks to correctly patch in new custom logics.

  - `ExtraHeaders/` - extra header files to interact with / describe types included in game binary that are not included in YRpp yet.
  - `Misc/` - uncategorized source code, including hooks that don't belong to an extension class.
  - `Utilities/` - common code that is used across the project.
  - `Phobos.cpp`/`Phobos.h` - extension bootstrapping code.
  - `Phobos.Ext.cpp` - contains common processing code new or extended classes. If you define a new or extended class you have to add your new class into `MassActions` global variable type declaration in this file.
- `YRpp/` - contains the header files to interact with / describe types included in game binary and also macros to write hooks using Syringe. Included as a submodule.

5.1.2 Code styleguide

We have established a couple of code style rules to keep things consistent. Some of the rules are enforced in `.editorconfig`, where applicable, so you can autoformat the code by pressing `Ctrl + K, D` hotkey chord in Visual studio. Still, it is advised to manually check the style before submitting the code.

- We use tabs instead of spaces to indent code.
- Curly braces are always to be placed on a new line. One of the reasons for this is to clearly separate the end of the code block head and body in case of multiline bodies:

```
if (SomeReallyLongCondition() ||
    ThatSplitsIntoMultipleLines())
{
    DoSomethingHere();
    DoSomethingMore();
}
```

- Braceless code block bodies should be made only when both code block head and body are single line, statements split into multiple lines and nested braceless blocks are not allowed within braceless blocks:

```
// OK
if (Something())
    DoSomething();

// OK
if (SomeReallyLongCondition() ||
    ThatSplitsIntoMultipleLines())
{
    DoSomething();
}

// OK
if (SomeCondition())
{
    if (SomeOtherCondition())
        DoSomething();
}

// OK
if (SomeCondition())
{
```

(continues on next page)

(continued from previous page)

```
    return VeryLongExpression()
        || ThatSplitsIntoMultipleLines();
}
```

- Only empty curly brace blocks may be left on the same line for both opening and closing braces (if appropriate).
- If you use if-else you should either have all of the code blocks braced or braceless to keep things consistent.
- Code should have empty lines to make it easier to read. Use an empty line to split code into logical parts. It's mandatory to have empty lines to separate:
  - `return` statements (except when there is only one line of code except that statement);
  - local variable assignments that are used in the further code (you shouldn't put an empty line after one-line local variable assignments that are used only in the following code block though);
  - code blocks (braceless or not) or anything using code blocks (function or hook definitions, classes, namespaces etc.);
  - hook register input/output.

```
// OK
auto localVar = Something();
if (SomeConditionUsing(localVar))
    ...

// OK
auto localVar = Something();
auto anotherLocalVar = OtherSomething();

if (SomeConditionUsing(localVar, anotherLocalVar))
    ...

// OK
auto localVar = Something();

if (SomeConditionUsing(localVar))
    ...

if (SomeOtherConditionUsing(localVar))
    ...

localVar = OtherSomething();

// OK
if (SomeCondition())
{
    Code();
    OtherCode();

    return;
}

// OK
if (SomeCondition())
{
    SmallCode();
    return;
}
```

- `auto` may be used to hide an unnecessary type declaration if it doesn't make the code harder to read. `auto` may not be used on primitive types.
- A space must be put between braces of empty curly brace blocks.
- To have less Git merge conflicts member initializer lists and other list-like syntax structures used in frequently modified places should be split per-item with item separation characters (commas, for example) placed *after newline character*:

```
ExtData(TerrainTypeClass* OwnerObject) : Extension<TerrainTypeClass>(OwnerObject)
    ,SpawnsTiberium_Type(0)
    ,SpawnsTiberium_Range(1)
    ,SpawnsTiberium_GrowthStage({ 3, 0 })
    ,SpawnsTiberium_CellsPerAnim({ 1, 0 })
{ }
```

- Local variables and function/method args are named in the `camelCase` (using a `p` prefix to denote pointer type for every pointer nesting level) and a descriptive name, like `pTechnoType` for a local `TechnoTypeClass*` variable.
- Classes, namespaces, class fields and members are always written in `PascalCase`.
- Class fields that can be set via INI tags should be named exactly like ini tags with dots replaced with underscores.
- Pointer type declarations always have pointer sign `*` attached to the type declaration.
- Non-static class extension methods faked by declaring a static method with `pThis` as a first argument are only to be placed in the extension class for the class instance of which `pThis` is.
  - If it's crucial to fake `__thiscall` you may use `__fastcall` and use `void* _` as a second argument to discard value passed through `EDX` register. Such methods are to be used for call replacement.
- Hooks have to be named using a following scheme: `HookedFunction_HookPurpose`, or `ClassName_HookedMethod_HookPurpose`. Defined-again hooks are exempt from this scheme due to impossibility to define different names for the same hook.
- Return addresses should use anonymous enums to make it clear what address means what, if applicable:

```
DEFINE_HOOK(0x48381D, CellClass_SpreadTiberium_CellSpread, 0x6)
{
    enum { SpreadReturn = 0x4838CA, NoSpreadReturn = 0x4838B0 };

    ...
}
```

- New ingame "entity" classes are to be named with `Class` postfix (like `RadTypeClass`). Extension classes are to be named with `Ext` postfix instead (like `RadTypeExt`).

Note: The styleguide is not exhaustive and may be adjusted in the future.

5.1.3 Git branching model

Couple of notes regarding the Git practices:

- We use git-flow-like workflow:
  - `master` is for stable releases, can have hotfixes pushed to it or branched off like a feature branch with the requirement of version increment and being merged into develop;
  - `develop` is the main development branch;
  - `feature/`-prefixed branches (sometimes the prefix may be different if appropriate, like for big fixes or changes) are so called "feature branches" - those are branched off `develop` for every new feature to be introduced into it and then merged back. We use squash merge to merge them back.
  - `hotfix/`-prefixed branches may be used in a same manner with `master` branch if needed, with a requirement of `master` being merged into `develop` after `hotfix/` branch was squash merged into `master`.
  - `release/`-prefixed branches are branched off `develop` when a new stable release is slated to allow working on features for a next release and stability improvements for this release. Those are merged with a merge commit into `master` and `develop` with a stable version increase, after which the stable version is released.
- When you're working with your local & remote branches use **fast-forward** pulls to get the changes from remote branch to local, **don't merge remote branch into local and vice versa**, this creates junk commits and makes things unsquashable.

5.2 Ways to help

Engine modding is a complicated process which is pretty hard to pull off, but there are also easier parts which don't require mastering the art of reverse-engineering or becoming a dank magician in C++.

5.2.1 Research and reverse-engineering

You can observe how the stuff works by using the engine and note which other stuff influences the behavior, but sooner or later you would want to see the innards of that. This is usually done using such tools as disassemblers/decompilers (IDA, Ghidra) to decipher what is written in the binary (`gamemd.exe` in case of the binary) and debuggers (Cheat Engine's debugger is pretty good for that) to trace how the binary works.

Hint: Reverse-engineering is a complex task, but don't be discouraged, if you want to try your hands at it ask us in the Discord channel, we will gladly help

---

Note: Assembly language and C++ knowledge, understanding of computer architecture, memory structure, OOP and compiler theory would certainly help.

5.2.2 Development

When you found out how the engine works and where you need to extend the logic you'd need to develop the code to achieve what you want. This is done by declaring a *hook* - some code which would be executed after the program execution reaches the certain address in binary. All the development is done in C++ using YRpp (which provides a way to interact with YR code and inject code using Syringe) and usually Visual Studio 2017/2019 or newer.

Contributing changes to the project

To contribute a feature or some sort of a change you you would need a Git client (I recommend GitKraken personally). Fork, clone the repo, preferably make a new branch, then edit/add the code or whatever you want to contribute. Commit, push, start a pull request, wait for it to get reviewed, or merged.

Hint: Every pull request push trigger a nightly build for the latest pushed commit, so you can check the build status at the bottom of PR page, press `Show all checks`, go to details of a build run and get the zip containing built DLL and PDB (for your testers, f. ex.), or download a build from an automatically posted comment.

Note: You'd benefit from C++ experience, knowledge of programming patterns, common techniques etc. Basic assembly knowledge would help to correctly write the interaction with the memory where you hook at. Basic understanding of Git and GitHub is also needed.

5.2.3 Testing

This is a job that any modder (and even sometimes player) can do. Look at a new feature or a change, try to think of all possible cases when it can work differently, try to think of any possible logic flaws, edge cases, unforeseen interactions or conditions etc., then test it according to your thoughts. Any bugs should be reported to issues section of this repo, if possible.

> **Warning:** **General stability** can only be achieved by extensive play-testing of new changes, both offline and online. Most modders have beta testing teams, so please, if you want the extension to be stable - contribute to that by having your testers play with the new features! Also the check-list below can help you identify issues quicker.

Testing check-list

- **All possible valid use cases covered**. Try to check all of the valid feature use cases you can think of and verify that they work as intended with the feature.
- **Correct saving and loading**. Most of the additions like new INI tags require storing them in saved object info. Sometimes this is not done correctly, especially on complex stuff (like radiation types). Please, ensure all the improvements work **identically** before and after being saved and loaded (on the same version of Phobos, of course).
- **Interaction with other features**. Try to use the feature chained or interacting with other features from vanilla or other libs (for example, mind control removal warhead initially was crashing when trying to remove mind control from a permanently mind-controlled unit).
- **Overlapping features not working correctly** (including those from third-party libs like Ares, HAres, CnCNet spawner DLL). Think of what features' code could overlap (in a technical sense; means they modify the same code) with what you're currently testing. Due to the nature of the project some features from other libs could

happen to not work as expected if they are overlapping (for example, when implementing mass selection filtering Ares' `GroupAs` was initially broken and units using it weren't being type selected properly).

- **Edge cases**. Those are the cases of some specific cases usually induced by some extreme parameter values (for example, vanilla game crashes on zero-size `PreviewPack` instead of not drawing it).
- **Corner cases**. Those are similar to edge cases but are hard to reproduce and are usually induced by a combination of extreme parameter values.

Note: Knowledge on how to mod YR and having an inquisitive mind, being attentive to details would help.

5.2.4 Writing docs

No explanation needed. If you fully understand how some stuff in Phobos works you can help by writing a detailed description in these docs, or you can just improve the pieces of docs you think are not detailed enough.

The docs are written in Markdown (which is dead simple, learn MD in 60 seconds; if you need help on extended syntax have a look at MyST parser reference). We use Sphinx to build docs, Read the Docs to host.

Hint: You don't need to install Python, Sphinx and modules to see changes - every pull request you make is being built and served by Read the Docs automatically. Just like the nightly builds, scroll to the bottom, press `Show all checks` and see the built documentation in the details of a build run.

There are two ways to edit the docs.

- **Edit from your PC**. Pretty much the same like what's described in contributing changes section; the docs are located in the `docs` folder.
- **Edit via online editor**. Navigate to the doc piece that you want to edit, press the button on the top right - and it will take you to the file at GitHub which you would need to edit (look for the pencil icon to the top right). Press it - the fork will be created and you'll edit the docs in your version of the repo (fork). You can commit those changes (preferably to a new branch) and make them into a pull request to main repo.

Note: OK English grammar and understanding of docs structure would be enough. You would also need a GitHub account.

5.2.5 Providing media to showcase features

Those would be used in docs and with a link to the respective mod as a bonus for the mod author. To record GIFs you can use such apps as, for example, GifCam.

Note: Please, provide screenshots, GIFs and videos in their natural size and without excess stuff or length.

5.2.6 Promoting the work

You can always help us by spreading the word about the project among people, whether you're an influential youtuber

CHAPTER

SIX

LICENSE

```
                GNU LESSER GENERAL PUBLIC LICENSE
                    Version 3, 29 June 2007
```

Copyright (C) 2007 Free Software Foundation, Inc. https://fsf.org/ Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public License, supplemented by the additional permissions listed below.

1. Additional Definitions.

As used herein, "this License" refers to version 3 of the GNU Lesser General Public License, and the "GNU GPL" refers to version 3 of the GNU General Public License.

"The Library" refers to a covered work governed by this License, other than an Application or a Combined Work as defined below.

An "Application" is any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library. Defining a subclass of a class defined by the Library is deemed a mode of using an interface provided by the Library.

A "Combined Work" is a work produced by combining or linking an Application with the Library. The particular version of the Library with which the Combined Work was made is also called the "Linked Version".

The "Minimal Corresponding Source" for a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version.

The "Corresponding Application Code" for a Combined Work means the object code and/or source code for the Application, including any data and utility programs needed for reproducing the Combined Work from the Application, but excluding the System Libraries of the Combined Work.

1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License without being bound by section 3 of the GNU GPL.

1. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or

b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

1. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.

b) Accompany the object code with a copy of the GNU GPL and this license document.

1. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.

b) Accompany the Combined Work with a copy of the GNU GPL and this license document.

c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.

d) Do one of the following:

```
0) Convey the Minimal Corresponding Source under the terms of this
License, and the Corresponding Application Code in a form
suitable for, and under terms that permit, the user to
recombine or relink the Application with a modified version of
the Linked Version to produce a modified Combined Work, in the
manner specified by section 6 of the GNU GPL for conveying
Corresponding Source.

1) Use a suitable shared library mechanism for linking with the
Library.  A suitable mechanism is one that (a) uses at run time
a copy of the Library already present on the user's computer
system, and (b) will operate properly with a modified version
of the Library that is interface-compatible with the Linked
Version.
```

e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

1. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.

b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

1. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.

CHAPTER

SEVEN

NEW / ENHANCED LOGICS

This page describes all the engine features that are either new and introduced by Phobos or significantly extended or expanded.

7.1 Buildings

7.1.1 Extended building upgrades logic

Upgrading own and allied Power Plants in CnC: Final War

- Building upgrades now can be placed on own buildings, on allied buildings and/or on enemy buildings. These

three owners can be specified via a new tag, comma-separated. When upgrade is placed on building, it automatically changes it's owner to match the building's owner.

- One upgrade can now be applied to multiple buildings via a new tag, comma-separated.
  - Currently Ares-introduced build limit for building upgrades doesn't work with this feature. This may change in future.

In `rulesmd.ini`:

```
[UPGRADENAME]       ; BuildingType
PowersUp.Owner=Self ; list of owners (Self, Ally and/or Enemy)
PowersUp.Buildings= ; list of BuildingTypes
```

7.2 Vehicles

7.2.1 Stationary vehicles

Setting VehicleType `Speed` to 0 now makes game treat them as stationary, behaving in very similar manner to deployed vehicles with `IsSimpleDeployer` set to true. Should not be used on buildable vehicles, as they won't be able to exit factories.

7.3 Technos

7.3.1 Mind Control enhancement

Mind Control Range Limit used in Fantasy ADVENTUREMultiple Mind Control unit auto-releases the first victim in Fantasy ADVENTURE

- Mind controllers now can have the upper limit of the control distance. Tag values greater than 0 will activate this feature.
- Mind controllers with multiple controlling slots can now release the first controlled unit when they have reached the control limit and are ordered to control a new target.
- Allows Warheads to play custom `MindControl.Anim` which defaults to `ControlledAnimationType`.

In `rulesmd.ini`

```
[SOMETECHNO]                        ; TechnoType
MindControlRangeLimit=-1.0          ; double
MultiMindControl.ReleaseVictim=no   ; boolean

[SOMEWARHEAD]                               ; Warhead
MindControl.Anim=ControlledAnimationType ; AnimType
```

7.3.2 Spawn range limit

Limited pursue range for spawns in Fantasy ADVENTURE

- The spawned units will abort the infinite pursuit if the enemy is out of range. `Spawner.ExtraLimitRange` adds extra pursuit range to the spawned units.

In `rulesmd.ini`:

```
[SOMETECHNO]               ; TechnoType
Spawner.LimitRange=no      ; boolean
Spawner.ExtraLimitRange=0  ; integer
```

7.3.3 Promoted Spawns

Promoted Spawns in Fantasy ADVENTURE

- The spawned units will promote as their owner's veterancy.

In `rulesmd.ini`:

```
[SOMETECHNO]               ; TechnoType
Promote.IncludeSpawns=no   ; boolean
```

7.3.4 Shield logic

Buildings, Infantries and Vehicles with Shield in Fantasy ADVENTURE

In `rulesmd.ini`:

```
[AudioVisual]
Pips.Shield=-1,-1,-1           ; int, frames of pips.shp for Green, Yellow, Red
Pips.Shield.Building=-1,-1,-1  ; int, frames of pips.shp for Green, Yellow, Red

[ShieldTypes]
0=SOMESHIELDTYPE

[SOMESHIELDTYPE]               ; ShieldType name
Strength=0                     ; integer
Armor=none                     ; ArmorType
Powered=false                  ; boolean
AbsorbOverDamage=false         ; boolean
SelfHealing=0.0                ; double, percents or absolute
SelfHealing.Rate=0.0           ; double, ingame minutes
Respawn=0.0                    ; double, percents or absolute
Respawn.Rate=0.0               ; double, ingame minutes
BracketDelta=0                 ; integer - pixels
IdleAnim=                      ; animation
IdleAnim.OfflineAction=Hides   ; AttachedAnimFlag (None, Hides, Temporal, Paused or␣
↪PausedTemporal)
IdleAnim.TemporalAction=Hides  ; AttachedAnimFlag (None, Hides, Temporal, Paused or␣
↪PausedTemporal)
BreakAnim=                     ; animation
HitAnim=                       ; animation

[SOMETECHNO]                   ; TechnoType
```

(continues on next page)

(continued from previous page)

```
ShieldType=SOMESHIELDTYPE          ; ShieldType; none by default

[SOMEWARHEAD]                   ; WarheadType
PenetratesShield=false          ; boolean
BreaksShield=false              ; boolean
```

- Now you can have a shield for any TechnoType. It serves as a second health pool with independent `Armor` and `Strength` values.
  - Negative damage will recover shield, unless shield has been broken. If shield isn't full, all negative damage will be absorbed by shield.
  - When the TechnoType with a unbroken shield, `[ShieldType]->Armor` will replace `[TechnoType]->Armor` for game calculation.
- When executing `DeploysInto` or `UndeploysInto`, if both of the TechnoTypes have shields, the transformed unit/building would keep relative shield health (in percents), same as with `Strength`. If one of the TechnoTypes doesn't have shields, it's shield's state on conversion will be preserved until converted back.
  - This also works with Ares' `Convert.*`.
- `Powered` controls whether or not the shield is active when a unit is running low on power or it is affected by EMP.
  - Attention, if TechnoType itself is not `Powered`, then the shield won't be offline when low power.
- `AbsorbOverDamage` controls whether or not the shield absorbs damage dealt beyond shield's current strength when the shield breaks.
- `SelfHealing` and `Respawn` respect the following settings: 0.0 disables the feature, 1%-100% recovers/respawns the shield strength in percentage, other number recovers/respawns the shield strength directly. Specially, `SelfHealing` with a negative number deducts the shield strength.
  - If you want shield recovers/respawns 1 HP per time, currently you need to set tag value to any number between 1 and 2, like `1.1`.
- `SelfHealing.Rate` and `Respawn.Rate` respect the following settings: 0.0 instantly recovers the shield, other values determine the frequency of shield recovers/respawns in ingame minutes.
- `IdleAnim`, if set, will be played while the shield is intact. This animation is automatically set to loop indefinitely.
  - `Bouncer=yes` animations are not supported at the moment.
- `IdleAnim.OfflineAction` indicates what happens to the animation when the shield is in a low power state.
- `IdleAnim.TemporalAction` indicates what happens to the animation when the shield is attacked by temporal weapons.
- `BreakAnim`, if set, will be played when the shield has been broken.
- `HitAnim`, if set, will be played when the shield is attacked, similar to `WeaponNullifyAnim` for Iron Curtain.
- A TechnoType with a shield will show its shield Strength. An empty shield strength bar will be left after destroyed if it is respawnable.
  - Buildings now use the 5th frame of `pips.shp` to display the shield strength while other units uses the 16th frame by default.

  - `Pips.Shield` can be used to specify which pip frame should be used as shield strength. If only 1 digit set, then it will always display it, or if 3 digits set, it will respect `ConditionYellow` and `ConditionRed`. `Pips.Shield.Building` is used for BuildingTypes.
  - `pipbrd.shp` will use its 4th frame to display an infantry's shield strength and the 3th frame for other units if `pipbrd.shp` has extra 2 frames. And `BracketDelta` can be used as additional `PixelSelectionBracketDelta` for shield strength.
- Warheads have new options that interact with shields.
  - `PenetratesShield` allows the warhead ignore the shield and always deal full damage to the TechnoType itself. It also allows targeting the TechnoType as if shield isn't existed.
  - `BreaksShield` allows the warhead to always break shields of TechnoTypes, regardless of the amount of strength the shield has remaining or the damage dealt, assuming it affects the shield's armor type. Residual damage, if there is any, still respects `AbsorbOverDamage`.

7.4 Weapons

7.4.1 Strafing aircraft weapon customization

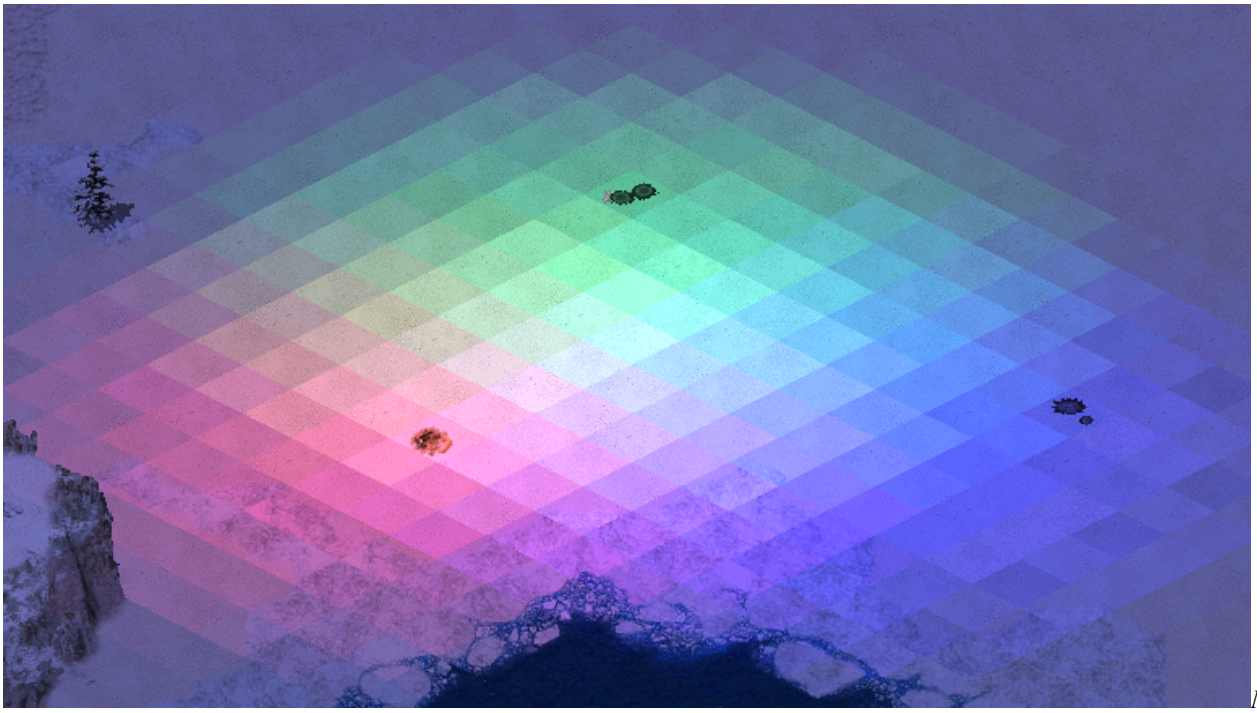
Strafing aircraft weapon customization in Project Phantom

- Some of the behavior of strafing aircraft weapons (weapon projectile has `ROT` below 2) can now be customized.
  - `Strafing.Shots` controls the number of times the weapon is fired during a single strafe run. `Ammo` is only deducted at the end of the strafe run, regardless of the number of shots fired. Valid values range from 1 to 5, any values smaller or larger are effectively treated same as either 1 or 5, respectively. Defaults to 5.
  - `Strafing.SimulateBurst` controls whether or not the shots fired during strafing simulate behavior of `Burst`, allowing for alternating firing offset. Only takes effect if weapon has `Burst` set to 1 or undefined. Defaults to false.

In `rulesmd.ini`:

```
[SOMEWEAPON]        ; WeaponType
Strafing.Shots=5             ; integer
Strafing.SimulateBurst=false ; bool
```

7.4.2 Custom Radiation Types



Mixing different radiation types

- Allows to have custom radiation type for any weapon now. More details on radiation here.

In `rulesmd.ini`

```
[RadiationTypes]
0=SOMERADTYPE

[SOMEWEAPON]                    ; WeaponType
RadType=Radiation               ; RadType to use instead
                                ; of default [Radiation]

[SOMERADTYPE]                   ; custom RadType name
RadDurationMultiple=1           ; int
RadApplicationDelay=16          ; int
RadApplicationDelay.Building=0  ; int
RadLevelMax=500                 ; int
RadLevelDelay=90                ; int
RadLightDelay=90                ; int
RadLevelFactor=0.2              ; double
RadLightFactor=0.1              ; double
RadTintFactor=1.0               ; double
RadColor=0,255,0                ; RGB
RadSiteWarhead=RadSite          ; WarheadType
```

7.4.3 Radiation enhancements

- Radiation now has owner by default, so any rad-kills will be scored. This behavior can be reverted by a corresponding tag.
  - `AffectsAllies`, `AffectsOwner` and `AffectsEnemies` on `RadSiteWarhead` are respected.
  - Currently the rad maker doesn't gain experience from kills, this may change in future.
- Radiation is now able to deal damage to Buildings. To enable set `RadApplicationDelay.Building` value more than 0.

In `rulesmd.ini`:

```
[SOMEWEAPON]    ; WeaponType
Rad.NoOwner=no  ; boolean
```

7.5 Warheads

Hint: All new warheads can be used with CellSpread and Ares' GenericWarhead superweapon where applicable.

7.5.1 Generate credits on impact

`TransactMoney` used in Rise of the East mod

- Warheads can now give credits to its owner at impact.

In `rulesmd.ini`:

```
[SOMEWARHEAD]   ; Warhead
TransactMoney=0 ; integer - credits added or subtracted
```

7.5.2 Reveal map for owner on impact

`SpySat=yes` on `[NUKE]` warhead reveals the map when nuclear missile detonates

- Warheads can now reveal the entire map on impact.
- Reveal only applies to the owner of the warhead.

In `rulesmd.ini`:

```
[SOMEWARHEAD] ; Warhead
SpySat=no     ; boolean
```

7.5.3 Shroud map for enemies on impact

- Warheads can now shroud the entire map on impact.
- Shroud only applies to enemies of the warhead owner.

In `rulesmd.ini`:

```
[SOMEWARHEAD] ; Warhead
BigGap=no     ; boolean
```

7.5.4 Remove disguise on impact

- Warheads can now remove disguise of spies.

In `rulesmd.ini`:

```
[SOMEWARHEAD]                    ; Warhead
RemoveDisguise=no                ; boolean
```

7.5.5 Break Mind Control on impact

- Warheads can now break mind control (doesn't apply to perma-MC-ed objects).

In `rulesmd.ini`:

```
[SOMEWARHEAD]                      ; Warhead
RemoveMindControl=no               ; boolean
```

7.5.6 Critical damage chance

- Warheads can now apply additional chance-based damage (known as "critical" damage) with the ability to customize chance, damage, affected targets, and animations of critical strike.

In `rulesmd.ini`:

```
[SOMEWARHEAD]       ; Warhead
Crit.Chance=0.0     ; float, chance on [0.0-1.0] scale
Crit.ExtraDamage=0  ; integer, extra damage
Crit.Affects=all    ; list of "affects" flags (same as SWType's)
Crit.AnimList=      ; list of animations

[SOMETECHNO]     ; TechnoType
ImmuneToCrit=no  ; boolean
```

7.5.7 Custom 'SplashList' on Warheads

- Allows Warheads to play custom water splash animations. See vanilla's Conventional system here.

In `rulesmd.ini`:

```
[SOMEWARHEAD]            ; Warhead
SplashList=<none>        ; list of animations to play
SplashList.PickRandom=no ; play a random animation from the list? boolean, defaults␣
↪to no
```

7.6 Projectiles

7.6.1 Projectile interception logic

Interception logic used in Tiberium Crisis mod

- Projectiles can now be made targetable by certain TechnoTypes. Interceptor TechnoType's projectile must be `Inviso=yes` and `AA=yes` in order for it to work properly and the projectile must be used in a primary Weapon.
  - `Interceptor.GuardRange` is maximum range of the unit to intercept projectile. The unit weapon range will limit the unit interception range though.
  - `Interceptor.EliteGuardRange` value is used if the unit veterancy is Elite.
  - `Interceptor.MinimumGuardRange` is the minimum range of the unit to intercept projectile. Any projectile under this range will not be intercepted.
  - `Interceptor.EliteMinimumGuardRange` value is used if the unit veterancy is Elite.

In `rulesmd.ini`:

```
[SOMETECHNO]                           ; TechnoType
Interceptor=no                         ; boolean
Interceptor.GuardRange=0.0             ; double
Interceptor.EliteGuardRange=0.0        ; double
Interceptor.MinimumGuardRange=0.0      ; double
Interceptor.EliteMinimumGuardRange=0.0 ; double

[SOMEPROJECTILE] ; Projectile
Interceptable=no ; boolean
```

7.7 Script actions

7.7.1 `71` Timed Area Guard

- Puts the TaskForce into Area Guard Mode for the given units of time. Unlike the original timed Guard script (`5,n`) that just stays in place doing a basic guard operation the "Area Guard" action has a more active role attacking nearby invaders or defending units that needs protection.

In `aimd.ini`:

```
[SOMESCRIPTTYPE]  ; ScriptType
x=71,n            ; integer, time in ingame seconds
```

7.7.2 72 Load Onto Transports

- If the TaskForce contains unit(s) that can be carried by the transports of the same TaskForce then this action will make the units enter the transports. In Single player missions the next action must be "Wait until fully loaded" (`43,0`) or the script will not continue.

In `aimd.ini`:

```
[SOMESCRIPTTYPE]  ; ScriptType
x=72,0
```

7.7.3 73 Wait until ammo is full

- If the TaskForce contains unit(s) that use ammo then the the script will not continue until all these units have fully refilled the ammo.

In `aimd.ini`:

```
[SOMESCRIPTTYPE]  ; ScriptType
x=73,0
```

CHAPTER

EIGHT

FIXED / IMPROVED LOGICS

This page describes all ingame logics that are fixed or improved in Phobos without adding anything significant.

8.1 Bugfixes and miscellaneous

- Fixed the bug when deploying mindcontrolled vehicle into a building permanently transferred the control to the house which mindcontrolled it.
- Fixed the bug when units are already dead but still in map (for sinking, crashing, dying animation, etc.), they could die again.
- Fixed the bug when cloaked Desolator was unable to fire his deploy weapon.
- SHP debris shadows now respect the `Shadow` tag.
- Allowed usage of TileSet of 255 and above without making NE-SW broken bridges unrepairable.

Side offset voxel turret in Breaking Blue project

- `TurretOffset` tag for voxel turreted TechnoTypes now accepts FLH (forward, lateral, height) values like `TurretOffset=F,L` or `TurretOffset=F,L,H`, which means turret location can be adjusted in all three axes.
- `InfiniteMindControl` with `Damage=1` can now control more than 1 unit.
- Aircraft with `Fighter` set to false or those using strafing pattern (weapon projectile `ROT` is below 2) now take weapon's `Burst` into accord for all shots instead of just the first one.
- `EMEffect` used for random AnimList pick is now replaced by a new tag `AnimList.PickRandom` with no side effect. (EMEffect=yes on AA inviso projectile deals no damage to units in movement)
- Script action `Move to cell` now obeys YR cell calculation now. Using `1000 * Y + X` as its cell value. (was `128 * Y + X` as it's RA leftover)
- The game now can reads waypoints ranges in [0, 2147483647]. (was [0,701])
- Map trigger action `125 Build At...` can now play buildup anim optionally (needs *following changes to `fadata.ini`*.
- Vehicles using `DeployFire` will now explicitly use weapon specified by `DeployFireWeapon` for firing the deploy weapon and respect `FireOnce` setting on weapon and any stop commands issued during firing.
- Fixed `DebrisMaximums` (spawned debris type amounts cannot go beyond specified maximums anymore). Only applied when `DebrisMaximums` values amount is more than 1 for compatibility reasons.
- Fixed building and defense tab hotkeys not enabling the placement mode after `Cannot build here.` triggered and the placement mode cancelled.
- Fixed building with `UndeployInto` plays `EVA_NewRallypointEstablished` while undeploying

Nod arty keeping target on attack order in C&C: Reloaded

- Vehicle to building deployers now keep their target when deploying with `DeployToFire`.

8.2 Technos

8.2.1 Customizable Teleport/Chrono Locomotor settings per TechnoType

Chrono Legionnaire and Ronco (hero) from YR:New War

- You can now specify Teleport/Chrono Locomotor settings per TechnoType to override default rules values. Unfilled values default to values in `[General]`.
- Only applicable to Techno that have Teleport/Chrono Locomotor attached.

In `rulesmd.ini`:

```
[SOMETECHNO]            ; TechnoType
WarpOut=                ; Anim (played when Techno warping out)
WarpIn=                 ; Anim (played when Techno warping in)
WarpAway=               ; Anim (played when Techno chronowarped by chronosphere)
ChronoTrigger=          ; boolean, if yes then delay varies by distance, if no it is␣
↪a constant
ChronoDistanceFactor=   ; integer, amount to divide the distance to destination by to␣
↪get the warped out delay
ChronoMinimumDelay=     ; integer, the minimum delay for teleporting, no matter how␣
↪short the distance
ChronoRangeMinimum=     ; integer, can be used to set a small range within which the␣
↪delay is constant
ChronoDelay=            ; integer, delay after teleport for chronosphere
```

8.2.2 Kill spawns on low power

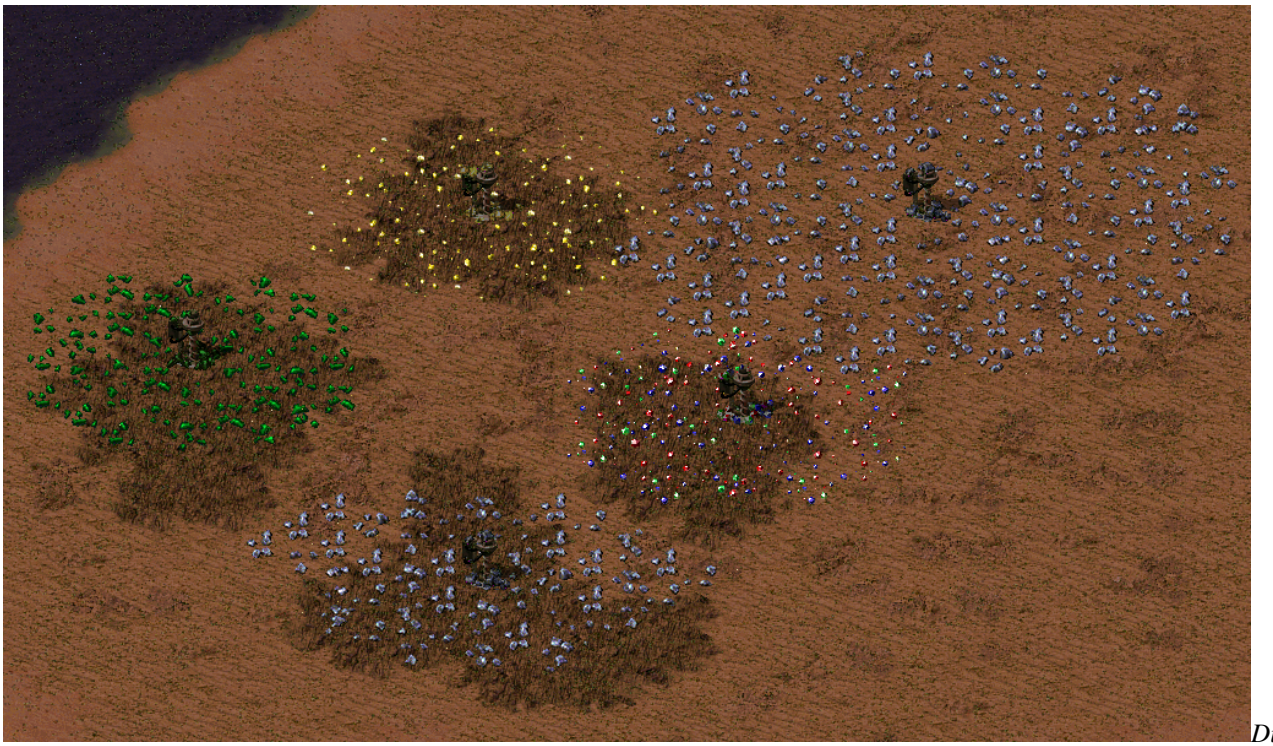
- `Powered=yes` structures that spawns aircraft like Aircrafts Carriers will stop targeting the enemy if low power.
- Spawned aircrafts self-destruct if they are flying.

In `rulesmd.ini`:

```
[SOMESTRUCTURE]       ; BuildingType
Powered.KillSpawns=no ; boolean
```

8.3 Terrains

8.3.1 Customizable ore spawners



Different ore spawners in Rise of the East mod

- You can now specify which type of ore certain TerrainType would generate.
- It's also now possible to specify a range value for an ore generation area different compared to standard 3x3 rectangle. Ore will be uniformly distributed across all affected cells in a spread range.
- You can specify which ore growth stage will be spawned and how much cells will be filled with ore per ore generation animation. Corresponding tags accept either a single integer value or two comma-separated values to allow randomized growth stages from the range (inclusive).

In `rulesmd.ini`:

```
[SOMETERRAINTYPE]              ; TerrainType
SpawnsTiberium.Type=0          ; tiberium/ore type index
SpawnsTiberium.Range=1         ; integer, radius in cells
SpawnsTiberium.GrowthStage=3   ; single int / comma-sep. range
SpawnsTiberium.CellsPerAnim=1 ; single int / comma-sep. range
```

8.4 Weapons

8.4.1 Customizable disk laser radius

- You can now set disk laser animation radius using a new tag.

In `rulesmd.ini`:

```
[SOMEWEAPON]           ; WeaponType
DiskLaser.Radius=38.2 ; floating point value
                       ; 38.2 is roughly the default saucer disk radius
```

8.4.2 Toggle-able ElectricBolt visuals

- You can now specify individual ElectricBolt bolts you want to disable. Note that this is only a visual change.

In `rulesmd.ini`:

```
[SOMEWEAPONTYPE]       ; WeaponType
IsElectricBolt=true    ; an ElectricBolt Weapon, vanilla tag
Bolt.Disable1=false    ; boolean
Bolt.Disable2=false    ; boolean
Bolt.Disable3=false    ; boolean
```

CHAPTER

NINE

USER INTERFACE

This page lists all user interface additions, changes, fixes that are implemented in Phobos.

9.1 Bugfixes and miscellanous

- Enabled ability to load full-color non-paletted PCX graphics of any bitness. This applies to every single PCX file that is loaded, including the Ares-supported PCX files.
- You can specify custom `gamemd.exe` icon via `-icon` command line argument followed by absolute or relative path to an `*.ico` file (f. ex. `gamemd.exe -icon Resources/clienticon.ico`).
- Fixed `Blowfish.dll`-caused error `***FATAL*** String Manager failed to initialize properly`, which occurred if `Blowfish.dll` could not be registered in the OS, for example, it happened when the player did not have administrator rights. With Phobos, if the game did not find a registered file in the system, it will no longer try to register this file, but will load it bypassing registration.

9.2 Audio

- You can now specify which soundtrack themes would play on win or lose.

In `rulesmd.ini`:

```
[SOMESIDE]              ; Side
IngameScore.WinTheme=   ; soundtrack theme ID
IngameScore.LoseTheme= ; soundtrack theme ID
```

9.3 Hotkey Commands

9.3.1 [ ] Next Idle Harvester

- Selects and centers the camera on the next TechnoType that is counted via the *harvester counter* and is currently idle.
- If need localization, just add `TXT_NEXT_IDLE_HARVESTER` and `TXT_NEXT_IDLE_HARVESTER_DESC` into your `.csf` file.

9.3.2 [ ] Dump Object Info

- Writes currently hovered or last selected object info in log and shows a message. See *this* for details.
- If need localization, just add `TXT_DUMP_OBJECT_INFO` and `TXT_DUMP_OBJECT_INFO_DESC` into your `.csf` file.

9.4 Battle screen UI/UX

9.4.1 Low priority for box selection

Harvesters not selected together with battle units in Rise of the East mod

- You can now set lower priority for an ingame object (currently has effect on units mostly), which means it will be excluded from box selection if there's at least one normal priority unit in the box. Otherwise it would be selected as normal. Works with box+type selecting (type select hotkey + drag) and regular box selecting. Box shift-selection adds low-priority units to the group if there are no normal priority units among the appended ones.

In `rulesmd.ini`:

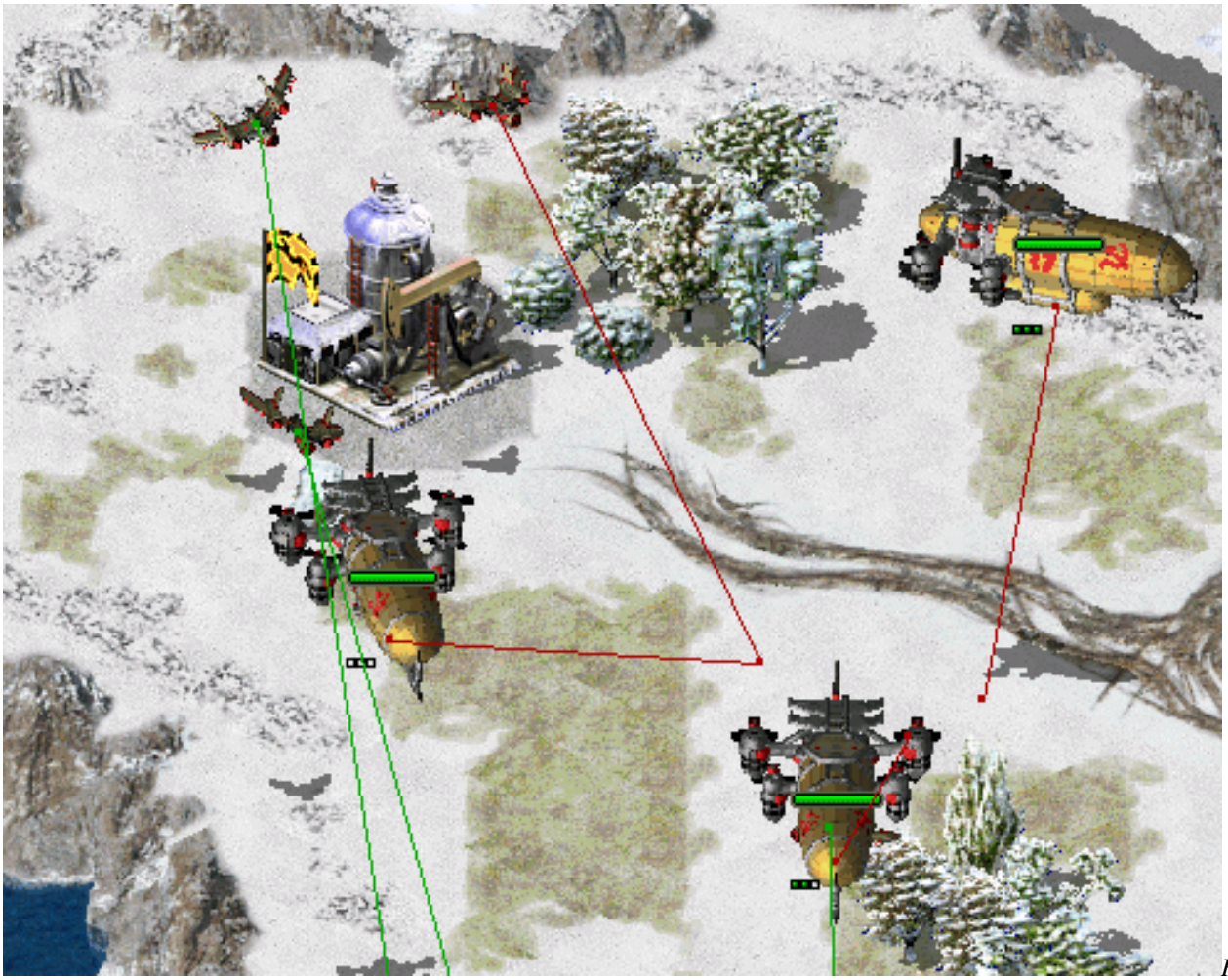
```
[SOMETECHNO]            ; TechnoType
LowSelectionPriority=no ; boolean
```

- This behavior is designed to be toggleable by users. For now you can only do that externally via client or manually.

In `RA2MD.ini`:

```
[Phobos]
PrioritySelectionFiltering=yes ; bool
```

9.4.2 Hide health bars



Health bars hidden in CnC: Final War

- Health bar display can now be turned off as needed, hiding both the health bar box and health pips.

In `rulesmd.ini`:

```
[SOMENAME]          ; TechnoType
HealthBar.Hide=no   ; boolean
```

9.5 Loading screen

- PCX files can now be used as loadscreen images.
  - You can specify custom loadscreen with Ares tag `File.LoadScreen`.
- The loadscreen size can now be different from the default `800x600` one; if the image is bigger than the screen it's centered and cropped.
  - This feature works in conjunction with CnCNet5 spawner DLL which resizes loadscreen window to actual monitor size and places the image in center. If there's no CnCNet5 spawner loaded, the window resolution will be always `800x600`.

- You can now disable hardcoded black dots that YR engine shows over empty spawn locations, which allows to use prettier and more correctly placed markers that are produced by Map Renderer instead.

In `uimd.ini`:

```
[LoadingScreen]
DisableEmptySpawnPositions=no ; boolean
```

9.6 Sidebar / Battle UI

9.6.1 Specify Sidebar style

- It's now possible to switch hardcoded sidebar button coords to use GDI sidebar coords.

In `rulesmd.ini`:

```
[SOMESIDE]            ; Side
Sidebar.GDIPositions= ; boolean
                      ; default values are:
                      ; yes for the first side
                      ; no for others
```

9.6.2 Custom Missing Cameo (`XXICON.SHP`)

- You can now specify any SHP/PCX file as XXICON.SHP for missing cameo.

In `rulesmd.ini`:

```
[AudioVisual]
MissingCameo=XXICON.SHP    ; filename - including the .shp/.pcx extension
```

9.6.3 Harvester counter

Harvester Counter in Fantasy ADVENTURE

- An additional counter for your active/total harvesters can be added near the credits indicator.
- You can specify which TechnoType should be counted as a Harvester. If not set, the techno with `Harvester=yes` or `Enslaves=SOMESLAVE` will be counted.
- The counter is displayed with the format of `Label(Active Harvesters)/(Total Harvesters)`. The label is `U+26CF` by default.
- You can adjust counter position by `Sidebar.HarvesterCounter.Offset`, negative means left/up, positive means right/down.
- By setting `HarvesterCounter.ConditionYellow` and `HarvesterCounter.ConditionRed`, the game will warn player by changing the color of counter whenever the active percentage of harvesters less than or equals to them, like HP changing with `ConditionYellow` and `ConditionRed`.

In `uimd.ini`:

```
[Sidebar]
HarvesterCounter.Show=no                ; boolean
HarvesterCounter.Label=<none>           ; CSF entry key
HarvesterCounter.ConditionYellow=99%    ; double, percentage
HarvesterCounter.ConditionRed=50%       ; double, percentage
```

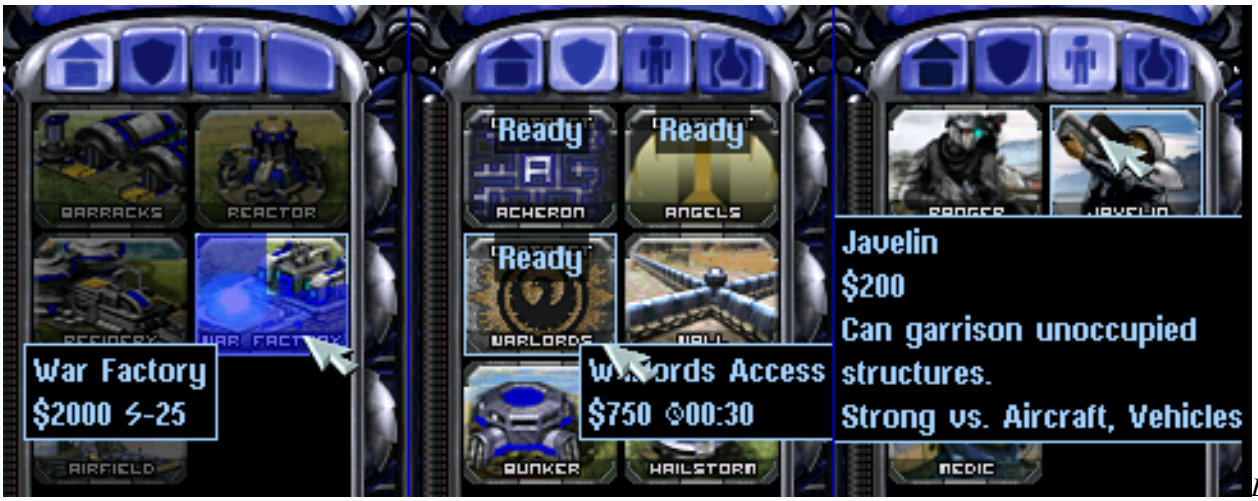
In `rulesmd.ini`:

```
[SOMETECHNO]        ; TechnoType
Harvester.Counted=  ; boolean
                    ; if set yes to a BuildingType like Oil Derricks
                    ; when producing cash, it will be counted as active

[SOMESIDE]                                      ; Side
Sidebar.HarvesterCounter.Offset=0,0             ; X,Y, pixels relative to default
Sidebar.HarvesterCounter.ColorYellow=255,255,0 ; R,G,B
Sidebar.HarvesterCounter.ColorRed=255,0,0       ; R,G,B
```

Note: If you use the vanilla font in your mod, you can use `the improved font` (v4 and higher) which among everything already includes the mentioned icons. Otherwise you'd need to draw them yourself using WWFontEditor, for example.

9.7 Tooltips



Extended tooltips used in CnC: Final War

- Sidebar tooltips can now display extended information about the TechnoType/SWType when hovered over it's cameo. In addition the low character limit is lifted when the feature is enabled via the corresponding tag, allowing for 1024 character long tooltips.
- TechnoType's tooltip would display it's name, cost, power and description (when applicable).
- SWType's tooltip would display it's name, cost, and recharge time (when applicable).
- Extended tooltips don't use `TXT_MONEY_FORMAT_1` and `TXT_MONEY_FORMAT_2`. Instead you can specify cost, power and time labels (displayed before correspoding values) with the corresponding tags. Characters `$` `U+0024`, `U+26A1` and `U+231A` are used by default.

- Fixed a bug when switching build queue tabs via QWER didn't make tooltips disappear as they should, resulting in stuck tooltips.
- The tooltips can now go over the sidebar bounds to accommodate for longer contents. You can control maximum text width with a new tag (paddings are excluded from the number you specify).

Note: Same as with harvester counter, you can download `the improved font` (v3 and higher) or draw your own icons.

In `uimd.ini`:

```
[ToolTips]
ExtendedToolTips=no ; boolean
CostLabel=<none>    ; CSF entry key
PowerLabel=<none>   ; CSF entry key
TimeLabel=<none>    ; CSF entry key
MaxWidth=0          ; integer, pixels
```

In `rulesmd.ini`:

```
[SOMENAME]           ; TechnoType or SWType
UIDescription=<none> ; CSF entry key
```

- The descriptions are designed to be toggleable by users. For now you can only do that externally via client or manually.

In `RA2MD.ini`:

```
[Phobos]
ToolTipDescriptions=yes ; bool
```

CHAPTER

TEN

MISCELLANOUS

This page describes every change in Phobos that wasn't categorized into a proper category yet.

10.1 Developer tools

10.1.1 Dump Object Info

Object info dump from CnC: Reloaded

- There's a new hotkey to dump selected/hovered object info on press. Available only when the debug tag is set.

In `rulesmd.ini`:

```
[GlobalControls]
DebugKeysEnabled=yes ; boolean
```

10.1.2 Semantic locomotor aliases

- It's now possible to write locomotor aliases instead of their CLSIDs in the `Locomotor` tag value. Use the table below to find the needed alias for a locomotor.

Alias	*CLSID*
Drive	`{4A582741-9839-11d1-B709-00A024DDAFD1}`
Jumpjet	`{92612C46-F71F-11d1-AC9F-006008055BB5}`
Hover	`{4A582742-9839-11d1-B709-00A024DDAFD1}`
Rocket	`{B7B49766-E576-11d3-9BD9-00104B972FE8}`
Tunnel	`{4A582743-9839-11d1-B709-00A024DDAFD1}`
Walk	`{4A582744-9839-11d1-B709-00A024DDAFD1}`
DropPod	`{4A582745-9839-11d1-B709-00A024DDAFD1}`
Fly	`{4A582746-9839-11d1-B709-00A024DDAFD1}`
Teleport	`{4A582747-9839-11d1-B709-00A024DDAFD1}`
Mech	`{55D141B8-DB94-11d1-AC98-006008055BB5}`
Ship	`{2BEA74E1-7CCA-11d3-BE14-00104B62A16C}`

CHAPTER

ELEVEN

PHOBOS

. . . is a WIP community project providing a set of new features and fixes for Yuri's Revenge based on modified YRpp and Syringe to allow injecting code. It's meant to accompany Ares rather than replace it, thus it won't introduce incompatibilities.

You can discuss the project at a dedicated channel on C&C Mod Haven.

11.1 Installation and Usage

1. If you don't have Syringe installed into your mod already, you can download it together with the latest Ares package. To install simply drop `Syringe.exe` into your game folder (where your `gamemd.exe` is located). It's highly recommended to **install Ares** too to get full Phobos feature set, just drop all the files from the archive except documentation folder into your game folder.
2. Obtain a Phobos "package" (official builds can be found on releases page; read below to learn how to get nightly builds). You should end up with two files: `Phobos.dll` and `Phobos.pdb`.
3. Place those files in the game folder (where your `gamemd.exe` is located).
4. To launch the game with Phobos (and all other installed Syringe-compatible engine extensions including Ares) you need to execute `Syringe.exe "gamemd.exe" [command line arguments for gamemd.exe]` in command line (omit arguments if you don't need any). `RunAres.bat` from Ares package does the same so you may use that as well.

If you already use Ares in your mod, you just need to drop Phobos files mentioned above in your game folder, Syringe will load Phobos automatically. This also applies to mods using XNA client with Syringe; if your mod doesn't use Syringe and Ares (or you just haven't set up the client) yet we recommend to use CnCNet client mod base by Starkku which is compatible with Ares and Phobos out of the box.

By default Phobos doesn't do any very noticeable changes except a few bugfixes. To learn how to use Phobos features head over to official documentation.

11.1.1 Obtaining nightly builds

For those who want to help testing Phobos features as soon as they are done - you can also get a nightly build. Those versions are bleeding edge (don't redistribute them outside of testing!) and have build information (commit and branch/tag) in them which is displayed ingame and can't be turned off. There are two ways to get a nightly build.

- **Get an artifact via nightly.link**. This is a service that allows guests to download automatic builds from GitHub. You can get a build for the latest successful (marked with a green tick) `develop` branch commit via this link, or get a build for any up-to-date pull request via an automatic comment that would appear in it.

- **Get an artifact manually from GitHub Actions runs**. You can get an artifact for a specific commit which is built automatically with a GitHub Actions workflow, just press on a green tick, open the workflow, find and download the build artifact. This is limited to authorized users only.

11.2 Building

1. Install **Visual Studio** (2019 is recommended, 2017 is minimum) with the dependencies listed in `.vswhere` (it will prompt you to install missing dependences when you open the project, or you can run VS installer and import the config). If you prefer to use **Visual Studio Code** you may install **VS Build Tools** with the stuff from `.vswhere` instead. You can also don't use any code editor or IDE and build via **command line scripts** included with the project.
2. Clone this repo recursively via your favorite git client (that will also clone YRpp).
3. To build the extension:

- in Visual Studio: open the solution file in VS and build it (`Debug` build config is recommended);
- in VSCode: open the project folder and hit `Run Build Task...` (`Ctrl + Shift + B`);
- barebones: run `scripts/build_debug.bat`.

1. Upon build completion the resulting `Phobos.dll` and `Phobos.pdb` would be placed in the subfolder identical to the name of the buildconfig executed.

11.3 Documentation

The documentation can be found at here @ Read the Docs and is split by a few major categories, each represented with a page on the sidebar. Each page has it's contents grouped into multiple subcategories, be it buildings, technotypes, infantries, superweapons or something else.

You can switch between versions in the bottom left corner, as well as download a PDF version.

11.3.1 How to read code snippets

```
; which section the entries should be in
; can be a freeform name - in this case the comment would explain what it is
; if no comment to be found - then it's a precise name
[SOMENAME]            ; BuildingType
; KeyName=DefaultValue ; accepted type with optional explanation
; if there's nothing to the right of equals sign - the default value is empty/absent
; if the default value is not static - it's written and explained in a comment
UIDescription=<none> ; CSF entry key
```

11.4 Credits

11.4.1 Developers

- **Belonit (Gluk-v48)** - project author (Patreon, PayPal)
- **Kerbiter (Metadorius)** - project co-author, project manager, DevOps, technical writer (Patreon)
- **Uranusian (Thrifinesma)** - developer, CN community ambassador (Patreon, AliPay)
- **secsome (SEC-SOME)** - developer (Patreon)
- **Otamaa (Fahroni, BoredEXE)** - developer (PayPal)
- **FS-21** - developer
- **Starkku** - developer

11.4.2 Contributions

- **Belonit (Gluk-v48)** - project creation, disable empty spawn positions, custom gamemd icon with Command Line, full-color non-paletted PCX, SpySat, BigGap, TransactMoney, PCX Loading Screen, custom DiskLaser radius, extended tooltips, building upgrades enhancement, hide health bar, Sidebar.GDIPosition, help with CellSpread, Blowfish.dll-related errors fix, zero size map previews, semantic locomotor aliases, shields
- **Kerbiter (Metadorius)** - SHP debris respect Shadow, building upgrades enhancement, extended tooltips, selection priority filtering, TurretOffset enhancement, customizable ore spawners, select next idle harvester hotkey, interceptor enhancement, zero size map previews, CI/CD, overhauled Unicode font, docs maintenance, VSCode configs, code style
- **tomsons26** - all-around help, assistance and guidance in reverse-engineering, YR binary mappings
- **CCHyper** - all-around help, current project logo, assistance and guidance in reverse-engineering, YR binary mappings
- **Ares developers** - YRpp and Syringe which are used, save/load, project foundation and generally useful code from Ares, unfinished RadTypes code, prototype deployer fixes
- **ZPHYUS** - win/lose themes code
- **ayylmao** - help with docs
- **SMxReaver** - help with docs, extensive and thorough testing
- **4SG** - help with docs
- **wiktorderelf** - overhauled Unicode font
- **Uranusian (Thrifinesma)** - Mind Control enhancement, custom warhead splash list, harvesters counter, promoted spawns, shields, death after dead fix, customizeable missing cameo, placement mode responding of tab hotkeys fix, overhauled Unicode font, help with docs
- **secsome (SEC-SOME)** - debug info dump hotkey, refactoring & porting of Ares helper code, introducing more Ares-derived stuff, disguise removal warhead, Mind Control removal warhead, Mind Control enhancement, shields, AnimList.PickRandom, MoveToCell fix, unlimited waypoints, Build At trigger action buildup anim fix, Undeploy building into a unit plays `EVA_NewRallyPointEstablished` fix
- **Otamaa (Fahroni, BoredEXE)** - help with CellSpread, ported and fixed custom RadType code, togglable ElectricBolt bolts, customizable Chrono Locomotor properties per TechnoClass, DebrisMaximums fixes
- **E1 Elite** - TileSet 255 and above bridge repair fix

- **FS-21** - Dump Object Info enhancements, Powered.KillSpawns, Spawner.LimitRange, ScriptType Actions 71, 72 & 73, MC deployer fixes, help with docs
- **AutoGavy** - interceptor logic, warhead critical damage system
- **ChrisLv_CN** - interceptor logic, general assistance
- **Xkein** - general assistance, YRpp edits
- **thomassneddon** - general assistance
- **Starkku** - Warhead shield penetration & breaking, strafing aircraft weapon customization, vehicle DeployFire fixes/improvements, stationary VehicleTypes
- **SukaHati (Erzoid)** - Minimum interceptor guard range
- **mevitar** - honorary shield tester *triple* award
- **Damfoos** - extensive and thorough testing
- **Rise of the East community** - extensive playtesting of in-dev features

Thanks to everyone who uses Phobos, tests changes and reports bugs! You can show your appreciation and help project by displaying the logo (monochrome version can be found here) in your client/launcher (make it open Phobos GitHub page for extra fanciness), linking to Phobos repository, contributing or donating to us via the links above.

11.5 Legal and License

The Phobos project is an unofficial open-source community collaboration project to extend the Red Alert 2 Yuri's Revenge engine for modding and compatibility purposes.

This project has no direct affiliation with Electronic Arts Inc. Command & Conquer, Command & Conquer Red Alert 2, Command & Conquer Yuri's Revenge are registered trademarks of Electronic Arts Inc. All Rights Reserved.