Breaking Changes

Introduction

Our goal is to keep breaking changes between any given version of X4 to an absolute minimum. In some cases, however, we have had to change things which break scripts/mods written for older versions.
The following table provides an overview about all breaking changes. This should help you to make any necessary changes to restore compatibility of older mods with new versions of X4.
Feel free to drop a note if a certain breaking change is breaking your mod and you have difficulties finding a way to work around the issue.

The list of breaking changes does **NOT** cover changes in the provided/shipped scripts (for instance UI scripts like the ones located under ui/addons/XXX or ui/core/Lua). These scripts can change anytime without explicit notice. If you hook into one of these scripts or provide replacements for these, please check the script for possible breaking changes yourself.

Further note that the list provides information about breaking changes which are in the pipeline of being released. This is meant merely as an informational heads up. Neither do mistake this as an announcement of that version becoming available soon nor take it for granted that such a change will go live. Any information provided for not yet released versions can change (and even be removed) prior to such version having seen the light of day.

As a final note, please be aware that issues introduced during the beta phase might not be explicitly mentioned in the list below, if things would only be broken in-between two beta versions. While we still aim to document any (potential) breaking change here, there might be circumstances for which we divert from that procedure (especially in case of only minor issues which are considered merely bugfixes rather than an intended behavior change).

UI modding considered unstable until further notice...
Please note that UI modding must be considered **unstable** in the current version. While we are working hard to get the UI modding integration into a stable state, we might have to introduce larger backwards incompatibilities in the following patches. This means that you might have to modify any mod using UI modding capabilities to a larger extend to keep it compatible with certain patches. Still, we are doing our best not to unnecessarily break things for modders and certainly will announce such changes on this page here.

Breaking Changes

Type	Version	Summary
3.20		
Scripts	3.20 Beta 1	Parameters of **<event_player_attacked_object/>** and **<event_object_attacked_object/>** changed
Old params: param = attacked object, param2 = attack method, param3 = attacked component *New params: param = attacked object, param2 = attack method, param3 = attack component detail list: [attacked component, attacking weapon]* *Note that this also means the component attribute to the events is no longer available, so scripts checking the attacked component need to use event.param3.{1} instead*		
Scripts	3.20 Beta 1	Removed script property $trade.**restriction.faction**
$trade.restriction.faction has been replaced with $trade.restriction.factions, which returns a list of factions.		
UI extensions	3.20 Beta 1	**Lua**: **GetTradeRestrictions()**, **ToggleFactionTradeRestriction()** and **ToggleFactionTradeWareOverride()** are now obsolete!
With the addition of the Trade Rule feature the given functions are obsolete and do not function anymore. This set of functions will return dummy values to avoid breaking scripts. Use GetAllTradeRules(), GetContainerTradeRuleID(), SetContainerTradeRule() and similar trade rule functions as replacement.		
3.10		
Scripts	3.10 Beta 1	Removed script action **<add_build/>**

Removed <add_build/> action which was no longer needed and would result in builds being added in unsupported ways. Use other <add_build_xxx/> actions.		

3.00

Scripts	3.0 Beta 6	Changed behaviour of script action **<get_suitable_job/>**
<get_suitable_job/> previously did not respect job quotas or the current number of job ships. It now does, only returning job IDs which are not exceeding the maxgalaxy quota. Using 'exceedquota' allows the action to behave as before.		
Scripts	3.0 Beta 6	Changed behaviour of script action **<get_ware_definition/>**
<get_ware_definition/> previously filtered for equipment wares by default. This is now not the case and existing cases now use flags="equipment". Additional filter attributes for group and tags has also been added.		
Global	3.0 Beta 5	Renamed all files in the **md**, ui, and **cutscenes** folders to lower-case
All MD, UI, and cutscene files have been renamed as lower-case. For most users this should have no effect since files packed in catalogs are essentially case-insensitive. However, this change can affect users on Linux systems when running a mod that contains single files instead of a catalog, and/or when playing with unpacked base game files. If your mod has XML patch files that are not packed in a catalog, the filenames must be lower-case to match the filenames in the base game. (For info on using mod catalogs, see the Readme of the X Catalog Tool.) *Note that this change has no effect on the actual content of scripts or cutscenes. The MD script/cutscene names are stored in the files, regardless of the filenames, so that references to them remain unchanged (e.g., "md.Setup.Start" references a cue in setup.xml).*		
Scripts	3.0 Beta 5	Adjustments of some MD library cues in the base game to change their usage from **<include_actions>** to <run_actions>
Some MD library cues in the base game that were previously used with <include_actions> have been adjusted, so they must now be used with the new MD action <run_actions> instead. Mods that use such a library with <include_actions> will have to switch to <run_actions> as well. A list of affected libraries is not provided here. Note: Additional library cues may be adjusted in future builds without further warning.		
Scripts	3.0 Beta 5	Changes to $dockingbay.**todockpos** and $dockingbay.**launchpos**
Before the change, positions are relative to $dockingbay.parent. *After the change, positions are relative to $dockingbay as specified in the documentation.*		
Global	3.0 Beta 2	Support for entity flag "**skillsvisible**" dropped
Entity skills are always visible; that flag was a leftover from XR. We removed support for the script property .skillsvisible, the attribute "skillsvisible" of <set_entity_traits> and <set_npc_template_traits>, and the parameter "skillsvisible" of Lua function GetComponentData().		
Scripts	3.0 Beta 2	Changes to **<event_object_changed_owner>** and added **<event_object_changed_true_owner>**.
Before 3.0 Beta 2 there was just a single <event_object_changed_owner> condition. This condition was triggered in multiple cases when an owner change occurred. However, the exact cases were inconsistent and also the event could have been triggered w/o an effective owner change having been applied. *3.0 Beta 2 fixes these inconsistencies and ensures that the event is only fired, if the "effective" component owner of the object changed. To handle cases where the script actually needs to be informed if the "true" owner of a component was changed (i.e. not taking the cover faction into account) a new <event_object_changed_true_owner> was introduced.* *If in your scripts you make use of the <event_object_changed_owner> condition you need to verify that you indeed want to work with the "effective" component owner or whether your script code actually should work with the true owner of an object and update your code accordingly.* *On top of that the 3rd parameter of <event_object_changed_owner> (trueprevious) was dropped as the event is no longer triggered upon a change of the true owner alone (i.e. only if the change of the true owner has also an effect on the "effective" component owner).*		
Global	3.0 Beta 1	**MissionBoard** support dropped
MissionBoards was a dummy asset type which was only used during early development and never meant to be shipped in the released version. If any mod tried to make use of this asset type, undefined behaviour would occur. Therefore we cleaned things up in 3.0 Beta 1 including deprecating/removing any related UI/script function.		
Job/God	3.0 Beta 1	By default, **job/god entries** now only spawn objects in space added by the extension in which they are defined
To better support extensions which add to the base game map, a job/god entry now defaults to only spawning objects in areas of the map that are added by the extension in which that entry is defined. You can override this behaviour by adding a matchextension="false" attribute to the job/god entry definition. This allows the entry to spawn objects anywhere that matches the entry's other criteria.		
Scripts	3.0 Beta 1	Script action **<add_actor_to_room/>** attribute **'room'** renamed to **'object'**
Due to engine changes, script action <add_actor_to_room/> has had the attribute 'room' changed to 'object', which is more accurate. Most likely requires a 'position' if a slot is not provided.		

Scripts	3.0 Beta 1	Script action **<set_doors_locked/>** attribute **'group'** changed
The 'group' attribute of <set_doors_locked/> can no longer be a list. It must be a single tag value. Previously, a list containing one tag was accepted, which was redundant.		
Scripts	3.0 Beta 1	Script conditions **<event_hack_*/>** and **<event_controlpanel_hack_*/>** removed
These events were never triggered, with the exception of <event_hack_started/> on the player entity for instant control panel hacks. Use <event_player_hacked_object> instead.		
Scripts	3.0 Beta 1	Script actions **<set_hack_target/>** and **<abort_hack/>** removed
These actions had no effect and were removed.		
Scripts	3.0 Beta 1	**<setup_conversation_minigame/>** script action removed
The script action was a leftover from the XR era and was never supported in X4 and couldn't be used in a meaningful way. It was therefore decided to better drop it altogether as it's not expected having been used by any mods.		
Scripts	3.0 Beta 1	<add_player_choice_*> **confidence** attribute removed
The default confidence of player options is another leftover from the XR era and had no effect in X4.		
Scripts	3.0 Beta 1	**<hack_via_control_panel/>** script action removed
Unused script action <hack_via_control_panel/> was removed.		
UI core	3.0 Beta 1	**Lua:** GetMiniGameCursorPosition() removed
GetMiniGameCursorPosition() was a left over from the XR era and never supposed to be shipped with X4. The function practically always returned 0 and hence we don't expect the removal of this obsolete function causing any actual mod to break.		
UI core	3.0 Beta 1	**FFI:** GetLocalizedInteractiveNotificationKey can return icon placeholders
In 3.0 Beta 1 we added support to display icons for keyboard bindings, if an icon is available for the specified keys. This results in GetLocalizedInteractiveNotificationKey() potentially returning a different text now if such an icon is assigned to the mapped key/button.		
UI extensions	3.0 Beta 1	**FFI:** UIWeaponMod returned by GetInstalledWeaponMod() now includes property SurfaceElementFactor
In 3.0 Beta 1 we added support for a new weapon mod property "SurfaceElementFactor" to allow weapons to deal increased damage to surface elements such as Shield Generators, Turrets or Engines.		
Scripts	3.0 Beta 1	**$ware.illegal** updated
$ware.illegal used to return true if the specified ware has the 'illegal' tag which is no longer used. *It now returns true if $ware is illegal to any faction in the game.*		
Scripts	3.0 Beta 1	MD script **RML_Flight_Along_Path** removed
The MD script RML_Flight_Along_Path was not referened and was removed.		
Scripts	3.0 Beta 1	parameters of **<event_venture_mission_completed/>** changed
old params: param = venture details, param2 = ships involved, param3 = duration *new params: param = venture detail list: [mission name, mission type], param2 = list of ships involved, param3 = duration*		
Scripts	3.0 Beta 1	**<set_object_wing_name/>** removed, **<set_object_fleet_name/>** added
The common action <set_object_wing_name/> has been replaced with <set_object_fleet_name/>		
Scripts	3.0 Beta 1	**$controllable.wing.*** removed, **$controllable.fleet.*** added
$controllable.wing.name, $controllable.wing.iscommander, and $controllable.wing.commander have been replaced with $controllable.fleet.name, $controllable.fleet.iscommander, and $controllable.fleet.commander		

2.60

Scripts	2.60 Beta 1	'**checkoperational**' filter behaviour changed

Actions and conditions which use the 'checkoperational' filter now behave differently. Instead of changing how 'class' and 'exactclass' behave, checkoperational == true adds an additional filter for the components being of state operational, equivalent to state="componentstate.operational". This is implicitly enabled in action elements (e.g. <find_.../>) or condition elements (e.g. <count_...>), meaning that find_ship will only find operational ships. Using such actions/conditions with checkoperational="false" will include non-operational components such as wrecked and constructions. Existing uses may find that results now exclude non-operational components. *Sub nodes such as match_child will have checkoperational default to false. Existing uses may find that results now include non-operational components.*		
Scripts	2.60 Beta 1	$container.**supplyresources** behavior changed
The script keyword $container.supplyresources now includes reserved wares.		

2.20

UI core	2.20 Beta 3/4	**Lua: GetControllerInfo()** returned mouseSteering/mouseCursor values are undefined.
2.20 Beta 3 introduced the new direct mouse steering mode. The mode was however not integrated as a first level mode and hence GetControllerInfo() returned the "gamepad" mode while direct mouse steering was active. *In 2.20 Beta 4 we improved the situation slightly so that a call to GetControllerInfo() will return either "mouseCursor" or "mouseSteering" in any of the 3 mouse modes.* *It's a pending change to ensure that the returned mode is more reasonable in an upcoming patch. So be aware that another breaking change might be introduced at some point.*		
Scripts	2.20 Beta 3	Attribute for build related conditions e.g. **<event_build_finished/>** renamed from 'buildmodule' to 'buildprocessor'
The underlying behaviour remains the same in that it involves the buildprocessor, not the buildmodule. The attribute name has simply been corrected.		

2.00

Scripts	2.00 Beta 1	**<event_build_finished/>** param2 now returns null instead of a construction sequence
The construction sequence which was finished should now be accessed via the buildtask provided via param3.		
Scripts	2.00 Beta 1	**param.boarding.{...}** strength parameters removed
The 'recruitstrength', 'veteranstrength' and 'elitestrength' script parameters were no longer required for balancing the boarding gameplay, and were removed.		
Scripts	2.00 Beta 1	**$defensible.boardee/$defensible.boarder** and **<set_object_boarder/>/<remove_object_boarder/>** removed
Due to changes to boarding in X4, the boarding connections accessed by these script properties and actions became redundent and thus, removed. Any connections set by these actions in a savegame will not survive loading.		
UI extensions	2.00 Beta 1	**FFI: GetUpgradeSlotCurrentComponent()/GetUpgradeSlotGroup()** works on non-operationals
Before 2.0 Beta 1 using GetUpgradeSlotCurrentComponent() or GetUpgradeSlotGroup() only performed on operational objects (i.e. not wrecked objects or objects under construction). Since this is inconsistent with the rest of the UI functions, this was considered a bug and fixed in 2.0 Beta 1. If you require the old behavior, use the FFI function: IsComponentOperational() to check the passed object's state before making the call.		
UI extensions	2.00 Beta 1	**FFI: SetFormationShape()** no longer indicates an error upon certain error cases
Before 2.0 Beta 1 a call to SetFormationShape() indicated an error state to the caller in certain cases where setting a formation shape (potentially) failed. Due to a design flaw in the handling of formations, this however doesn't do any good, since the function by itself doesn't really set the formation shape in all cases. On top of that the call can fail at random. Hence, at the moment an indication of an error case that setting a formation potentially fails is no good to the caller and he has no means to distinguish that case from a real error case. *Therefore, in 2.0 Beta 1 we drop the error indication in case of an attempt to set the formation fails. We are currently working on a better solution to the underlying problem and hope to have it ready as part of a following beta.*		
Scripts	2.00 Beta 1	**<create_formation/>** now requires the attributes: 'leader' and 'follower'. Attribute 'object' is now deprecated.
This now enforces a leader and at least one follower in order to create or change a formation. Former implementation made it possible to create formations without any followers which led to issues.		
Scripts	2.00 Beta 1	**<event_player_changed_target/>** now fires and returns null if the player deselects a target.

For prior behavior, check for event.param being non-null.		
AI Scripts	2.00 Beta 1	**<shoot/>/<shoot_at/>** attribute changes.
Changed: Attribute "primary" is now optional and defines whether weapons from the ship's active primary weapon group will be fired. Defaults to true. *New attribute: "secondary" is an optional attribute that defines whether weapons from the ship's active secondary weapon group will be fired. Defaults to true.* *New attribute: "missiles" is an optional attribute that defines whether only missile launchers will be used. If false, only guns will be fired. Defaults to false.* *Corrects now-obsolete assumption that primary weapon groups only contain guns and secondary weapon groups only contain missile launchers.*		
UI extensions	2.00 Beta 1	**FFI: GetBuildTask()/GetNumBuildTasks()** got a new "buildmoduleid" parameter.
Added possibility to query build tasks of a certain buildmodule.		
UI extensions	2.00 Beta 1	**Lua: GetLibraryEntry()** retrieved a new "buildresources" field.
Added list of resources a buildmodule needs to build.		
UI extensions	2.00 Beta 1	**Lua: CalculateTotalHullFraction()** was removed.
Unused, derelict function from XR - use GetComponentData(..., "hullpercent") instead.		
UI extensions	2.00 Beta 1	**FFI: RequestDockAtReason()/UndockPlayerShip()** changed their return values.
Changed return value type to const char to better identify failure reasons in Lua script.*		

1.50

UI extensions	1.50 Beta 3	**Lua: GetComponentData()** changed behavior of "docksizes" propertly.
The "docksizes" property no longer includes internal ship storage docks.		
All	1.50 Beta 2	**Meaning of "Shipyard"** was corrected throughout Lua/MD/AI scripts, so that Wharfs are not Shipyards (unlike in XR).
Shipyards (formerly also known as Capital Shipyards) can build capital ships, while Wharfs only build non-capital ships. To check whether a station is a Shipyard or a Wharf, check whether the station "can build ships". *Lua: Adjusted GetComponentData() and HasShipyard(), added HasWharf()* *Lua / GetComponentData() properties: Removed "iscapitalshipyard", fixed "isshipyard", added "canbuildships", "iswharf", "isequipmentdock"* *Scripts: Fixed property .isshipyard, added .iswharf (.canbuildships existed already)*		
UI extensions	1.50 Beta 2	**FFI: UpgradeGroupInfo** datatype was changed.
Added new "operational" field to retrieve information about number of operational upgrades in a group.		
UI extensions	1.50 Beta 1	**FFI: SetGuidance()** removed useinfopoint argument.
In X Rebirth "info points" existed (which were mainly used as interaction points to scan stations/access information about modules). These were removed in X4 but this left-over property here was missed to be removed in-time for release. It was therefore removed now in order to move towards a stable/clean UI API. *The new function declaration is: void SetGuidance(UniverseID componentid, UIPosRot offset)*		
UI extensions	1.50 Beta 1	**FFI: CancelConstruction()** changed its return value.
CancelConstruction() now returns if the cancellation was successful.		

1.32

UI extensions	1.32	**Lua: GetComponentData()** removed the "nextdestname" property.
Instead of "nextdestname", use the properties "destination" or "destinationsector" to retrieve a destination component, then retrieve its "name" property.		

1.20

UI extensions	1.20	**FFI: GetAAOption()** got a new "useconfig" parameter.
The AA setting was changed to check the new setting with the user. In order to restore the old setting GetAAOption() gets the option to retrieve the config settings.		