# Kodu Localization

By Stephen Coy (KoduSupport@infiniteinstant.com)

With each new release we're trying to make it easier to localize Kodu so that it’s accessible to more people world-wide. All of Kodu’s translations have been done by volunteer community members. When we do a release of Kodu we include the latest versions of all the localized files but we also have a mechanism in place which allows us to update the localized files without user intervention. When Kodu starts up it pings out localization server to get a list of the available languages and the dates that those languages have been most recently updated. If the local versions of the files are older than the ones available online, Kodu downloads the updated versions.

The downloaded files are stored in the folder :

C:\Users\%USERNAME%\Documents\SavedGames\Boku\Player1\Content\Xml\Localizable

If you look in this folder there will be a file called Locales.Xml and one or more sub-folders. The Locales.Xml file contains the list of available languages and the time of their latest updates. You will also see a folder EN which contains the English Xml files. If you have run Kodu in any language other than English there will be a folder for that language. The list of languages in this file are the languages that will appear in the Language dialog in the Options Menu in Kodu.

The folder names come from the 2-letter language code as specified by ISO 639-1. You can get the full list of codes here: <https://en.wikipedia.org/wiki/List_of_ISO_639-1_codes>

The first step in adding a new language to Kodu is to add an entry to the Locales.Xml file for the language. The editor Notepad that comes with Windows is a good choice for editing this file since it works well with all fonts. Each entry contains four pieces of information:

* Language – This is the name of the language in English.
* Directory – This is the 2 letter ISO 639-1 code for the language which will also be used as the directory name.
* Native – This is the name of the language in that language.
* LastUpdated – This is the time and date this language was last updated. When adding a new language the date isn’t important except it should be some time in the past.

The easiest way to add a new language is just to copy one of the existing Locale blocks and change the first three entries.

After changing Locales.Xml you then need to create a new folder and populate it with the English files. One way to do this is to use Windows Explorer: Go to the Localization folder, right-click on the EN folder, and choose Copy. Then right click on a blank area of the Localization folder and choose Paste. You can then rename “EN – Copy” to the proper 2-letter language code.

At this point you should be able to run Kodu, go into the Options Menu and choose your new language. Of course everything will still be in English. To verify it’s working, close Kodu and edit the Strings.Xml file in your language’s folder. In that file, scroll down to ~line 95 and you will find the strings associated with the Main Menu. As a test, translate one or two of these strings, save the file, and then run Kodu. You should see your translated strings on the Main Menu. If all this works you are ready to start translating in earnest. If you have any trouble with any of this please email me so we can figure out what the problem is. My email is at the top of the document.

A word of caution. It’s very easy to make a mistake hand editing Xml files. Sometimes a mistake can cause the files to not load correctly which will cause Kodu to fall back to the English versions or crash. So if you know you’ve changed something but it still shows up in English then this is probably what is happening.

It is often helpful to regularly back up your changes. The simplest way to do this is to create a “backup” folder in the Localization folder and regularly copy/paste your language folder into the backup folder. Just be sure to verify the changes work *before* backing them up.

If something completely gets broken you can zip up the whole Localization folder and email it to me. I should be able to untangle whatever when wrong.

Even though there are seven files to translate the most important ones are Strings.Xml and Cards.Xml so you may want to attack those first. Strings.Xml contains the string used in the menus and dialogs. Cards.Xml contains the names and descriptions of the programming tiles. HelpOverlays.Xml is probably the next best candidate. It contains the help text which is displayed in the upper left-hand corner of the screen.

When Kodu is started in a non-English language Kodu writes a text file detailing the missing or unlocalized strings. This file is written to My Documents\SavedGames\Boku\Player1 and is called MissingLocalization.txt. As Kodu starts up this file will be populated with a list of possible problems with the translated .Xml files for the current language. This file will list all of the localized strings that are either missing from the localization directory or that are identical to the English version. This isn’t useful for adding a new language but can be of help which finding strings that you still need to localize or new strings that we’ve added since the previous version. Kodu creates this file by comparing the values in the localized files with their English defaults: if a value in the English translation has no counterpart in the localized files, an entry labeled "missing" will be added to the MissingLocalization.txt file; if a value in the English translation is thought to be equal to its counterpart in the localized files, an entry labeled "identical" will be added to the MissingLocalization.txt file.

Below are examples of what needs to be translated for each file (values needing translation are underlined). Only change the indicated strings.

Cards.Xml – This contain the info for the programming tiles. The “label” string is the label printed onto the bottom of the programming tile. The longer string is the description displayed by the tool tips.

|  |
| --- |
| <filter>  <red label="Red">React to something red.</red>  </filter> |

Help.Xml – This contain the example code used when a user chooses help while programming or in the Add Item menu. The only thing that should be changed is the description string.

|  |
| --- |
| <programmingExamples>  <ExamplePage>  <description>When I see an Apple, move toward it.</description>  <reflexes>  …  </reflexes>  </ExamplePage>  </programmingExamples> |

HelpOverlays.Xml – The Help Overlays are the help text which appears in the upper left hand corner of the screen telling the user what options are currently available. The id should **not** be changed. This particular file has a lot of duplicate entries so you can take advantage of search and replace.

|  |
| --- |
| <overlay>  <id>MiniHub</id>  <a>Select</a>  <start>Resume</start>  <bottom>Press &lt;start&gt; at any time to return here.</bottom>  <keyMouse>  [enter] Select  </keyMouse>  <keyMouseBottom>Press [home] at any time to return here.</keyMouseBottom>  </overlay> |

Strings.Xml – This contains the majority of the strings in Kodu. These are the strings associated with all the menus and the common strings used throughout the game such as the character names. The section marked buttonAliasStrings should not be translated. These are the strings used to trigger the embedded icons and need to stay unchanged.

|  |
| --- |
| <actorNames>  <pushPad>Push Pad</pushPad>  </actorNames> |

TutorialCrumbs.Xml – This is an obsolete file which is only here so that older versions of Kodu still function. It contains the text associated with the tutorial system. Only the entries marked gamepadText or mouseText should be changed. (The touchText entries are not functional on older systems so it makes no sense to translate them).

|  |
| --- |
| <Crumb>  <curModes>  <mode>InGame</mode>  </curModes>  <targetMode>MouseCameraMove</targetMode>  <gamepadText>Press &lt;back&gt; to go to Edit mode.</gamepadText>  <mouseText>Press [esc] to go to Edit mode.</mouseText>  <touchText>Tap [here] to go to Edit mode.</touchText>  </Crumb> |

TutorialStrings.Xml – This is the current text associated with the tutorial system. Only the entries marked gamepadText, mouseText, or touchText should be changed.

|  |
| --- |
| <Crumb id=”MouseCameraMove0”>  <gamepadText>Press &lt;back&gt; to go to Edit mode.</gamepadText>  <mouseText>Press [esc] to go to Edit mode.</mouseText>  <touchText>Tap [here] to go to Edit mode.</touchText>  </Crumb> |

TweakScreenHelp.Xml – This is the help text associated with the Options Menu, the Object Settings and the World Settings menus. Here again the id should be left alone.

|  |
| --- |
| <helpText id="Invulnerable">  <desc>Makes this bot indestructible.</desc>  </helpText> |

There will obviously be some entries in this that don’t really need attention, for instance we really don’t expect anyone to localize the name “Kodu”. During the running of Kodu, if any strings are missing from the localization, Kodu will default to using the English version of those strings.

When done with translating the files, in whole or partially, you can email the files to me and I can add them to the localization server which will make them available to everyone. Because the server is so easy to update please don’t hesitate to send even minor updates if needed.