

Lab 9: Appendix A Installing Qt IFW

- READ EACH STEP CAREFULLY -

Note: This guide only covers Qt5 installations using **Mingw** on **Windows**.

Note 2: You may also find the ***Tutorial*** on Installer Framework more useful than this lab write-up. <https://doc.qt.io/qtinstallerframework/ifw-tutorial.html>

Download and install Qt IFW: <https://wiki.qt.io/Qt-Installer-Framework>

1. Download the latest Qt IFW prebuilt (or build yourself).
 - a. <http://download.qt.io/snapshots/ifw/installer-framework/19/>
 - b. Extract the ifw-bld folder and all its contents into Qt install directory. (**C:/Qt** - *often*)
 - c. Open the ifw-bld directory and then navigate and open 'docs/html'.
 - d. Take notice of the extensive help and documentation files contained.

The rest of the lab (except deployment) is not specific to one system .

Lab 9: Appendix B Creating Your Config.xml

- READ EACH STEP CAREFULLY -

Create Folder Structure and config.xml:

1. Inside your source directory create the following folder structure:
 - a. **NOTE** for com.vendor.app use **com.(Organization).(ApplicationName)**
 - i. Example: **com.oit.blasteroids** or **com.cst238.mediaplayer**

Installer // Create this directory in the same directory as your project file

|----- config

|----- packages

|----- com.vendor.app

|----- meta

|----- data

2. Create a file in the **config** directory called '*config.xml*'. **Add contents:**

```
<?xml version="1.0" encoding="UTF-8"?>
<Installer>
  <Name>Your application</Name>
  <Version>1.0.0</Version>
  <Title>Your application Installer</Title>
  <Publisher>Your vendor</Publisher>
  <StartMenuDir>Super App</StartMenuDir>
  <TargetDir>@HomeDir@/InstallationDirectory</TargetDir>
</Installer>
```

Fill in with your application information.

Read more here : <https://doc.qt.io/qtinstallerframework/ifw-globalconfig.html>

Lab 9: Appendix C Creating Your Package

- READ EACH STEP CAREFULLY -

Create package.xml:

Packages are “installable” components. Each package is described using a package.xml file. The actual files to be installed will be put into the **data** directory. The *package.xml* that describes the package should be put into **meta** directory.

Note: Multiple packages are great but your installer can use one package to install the entire game.

1.) Under your package/meta directory make the **package.xml** file. **Add Contents:**

```
<?xml version="1.0" encoding="UTF-8"?>
<Package>
  <DisplayName>The root component</DisplayName>
  <Description>Install this example.</Description>
  <Version>0.1.0-1</Version>
  <ReleaseDate>2010-09-21</ReleaseDate>
  <Licenses>
    <License name="Beer Public License Agreement" file="license.txt" />
  </Licenses>
  <Default>script</Default>
  <Script>installscript.qs</Script>
  <UserInterfaces>
    <UserInterface>page.ui</UserInterface>
  </UserInterfaces>
</Package>
```

But fill in your application information. Key items - **<User Interfaces>** describes the pages you will have for this package. Similar **<Script>** is specific to the package. An example, a package to handle *shortcuts* (desktop and startmenu) would describe the “how” to check and generate those shortcuts in the **installscript.qs**.

Package.xml file reference -

<https://doc.qt.io/qtinstallerframework/ifw-component-description.html#package-information-file-syntax>

Read <https://doc.qt.io/qtinstallerframework/ifw-component-description.html#data-directory>
then continue to step 2.

2.) Place the *contents* for the package in the **data** directory. What is content? Likely if you have one component, this includes EVERY file necessary for running the application. Libraries, binaries, assets (images, music, etc). For more information on this part - see “**Packaging binaries for deploy**”.

3.) If you have install scripts they should be placed in package/**meta**.

Ultimately this is the end of creating the installer - understand the installer can get very complex depending on what you add. If you would like to improve continue reading: **Creating a Package Information File** section on the tutorial link on page 1.

4.) Repeat steps 1-3 over and over until all components/packages are added.

Lab 9: Appendix D Creating Installer Pages

- READ EACH STEP CAREFULLY -

Creating Pages:

If you have not discovered yet, Qt has a Designer tool. Using the designer tool you can generate pages for your installer.

<http://doc.qt.io/qtinstallerframework/ifw-customizing-installers.html>

1.) Open the Qt Designer tool:

C:/Qt/<Qt Version>/<Qt installation>/bin/designer.exe

2.) Choose **widget** from the dialog type on the left and then click **create**.

3.) Use the designer tool to build your UI file.

a.) <http://doc.qt.io/qt-5/qtdesigner-manual.html>

b.) <http://doc.qt.io/qt-5/gettingstartedqt.html>

4.) Save the .ui file with an appropriate name into your package/**meta** directory.

5.) Modify your **package.xml** file and add a **Userinterfaces** section. Then add a section to include the newly created .ui file.

```
<UserInterfaces>
```

```
    <UserInterface> yourwidget.ui</UserInterface>
```

```
</UserInterfaces>
```

If you are lost on <tags> re-read the section on package information file syntax

<http://doc.qt.io/qtinstallerframework/ifw-component-description.html#data-directory>

6.) If you have not added a qt script to your package, add it now

packages/meta/**package.qs**

7.) Inside package.qs, Add the Component section into the script and the necessary lines to add your new .ui page

```
function Component()  
{ // this is your package constructor  
    component.loader.connect(this, Component.prototype.loaded); // make loaded connection  
    if(!installer.addWizardPage(component, "YourWidgetName", QInstaller.ReadyForInstallation))  
        console.log("Could not load my widget");  
}
```

Then later in the script you can do stuff when the user has entered your widget

```
Component.prototype.yourWidgetNameEntered = function ()  
{  
    var pageWidget = gui.pageWidgetByObjectName("YourWidgetName"); // handle to the widget  
}
```

8.) Read a lot more about scripting options, check out the examples.

a.) Scripting - <http://doc.qt.io/qtinstallerframework/ifw-customizing-installers.html>

b.) Examples - <http://doc.qt.io/qtinstallerframework/qtifwexamples.html>

9.) Finish your installer.

- a.) Add more packages if you would like.
- b.) Add shortcuts - startmenu or desktop if you want
- c.) Add anything you want to to customize your installer.

Lab 9: Appendix E Generating Installer

- READ EACH STEP CAREFULLY -

Locate Tools and Source Directories:

- 1.) Find the exact directory that **your project is located** in - this is absolute path
 - a.) example: **C:/Documents/MyProjectDir/src** - assuming *src* directory contains *<project_name>.pro*
 - b.) Copy the complete path to that directory into notepad (or any text file).
 - c.) Going forward this directory path will be referenced as **<SRC_DIR>**
- 2.) Locate the exact directory that your **ifw tools** are located in (bin folder) - this is an *absolute path*
 - a.) example: **C:/Qt/ifw-bld/bin** -assuming this directory contains *binarycreator.exe*
 - b.) Copy this path to a directory into notepad (or any text file).
 - c.) Going forward this directory path will be referenced as **<TOOL_DIR>**

*****Time to Start Windows Deployment:*****

<http://doc.qt.io/qt-5/windows-deployment.html>

Creating a “release” of your project:



- 1.) Under **Qt** in your start-menu or (**cmd.exe** and run **qtenv2.bat** in **C:/Qt/<version>/mingw/bin**) open the **Qt command line environment**.

- 2.) Then navigate to your **source directory**. Use your path for **<SRC_DIR>**.

```
C:\Users\Christopher>C:\Qt\5.6\mingw49_32\bin\qtenv2.bat
C:\Users\Christopher>echo off
Setting up environment for Qt usage...
C:\Qt\5.6\mingw49_32>cd <SRC_DIR>_
```

- 3.) To clean your directory run **mingw32-make clean**. If your makefile doesn't exist this will fail (don't worry that is fine).

4.) Now generate your project makefile in release mode using the *qmake* tool.