



bada Tutorial:
Development Environment
(bada Developer Site, SDK, UI
Builder, Testing Tool)

Contents

- From Conception to Release
- Application Development Process
- Developer Site
- SDK
- UI Builder
- bada Testing Tool
- FAQ



From Conception to Release

The bada SDK and tools provide everything you need to manage your application life-cycle from product conception, through development, release, and end-of-life application retirement.



Application Development Process

1 Developer Site (developer.bada.com)

- Sign up
- Generate a new application profile
- Set application version
- Select privileged groups to use
- Set up bada components
- Select application features



• Remove application

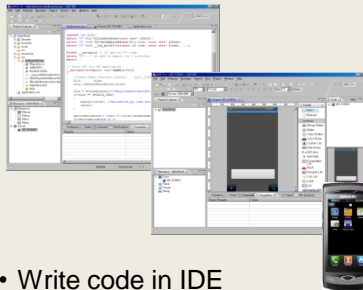
Download SDK

Download manifest.xml

SDK

2

- Run Application Wizard



- Write code in IDE
- Design UI with UI Builder
- Build, debug, and test with Simulator
- Build, debug, and test on device
- Create package and deploy

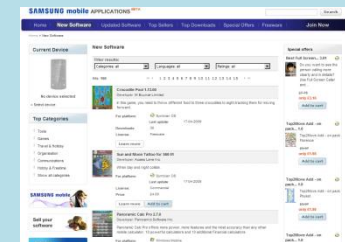
3

Registration for Certification

Application Development Process:

- 1 Create your application ID.
- 2 Write your application.
- 3 Test your application.
- 4 Sell your application.

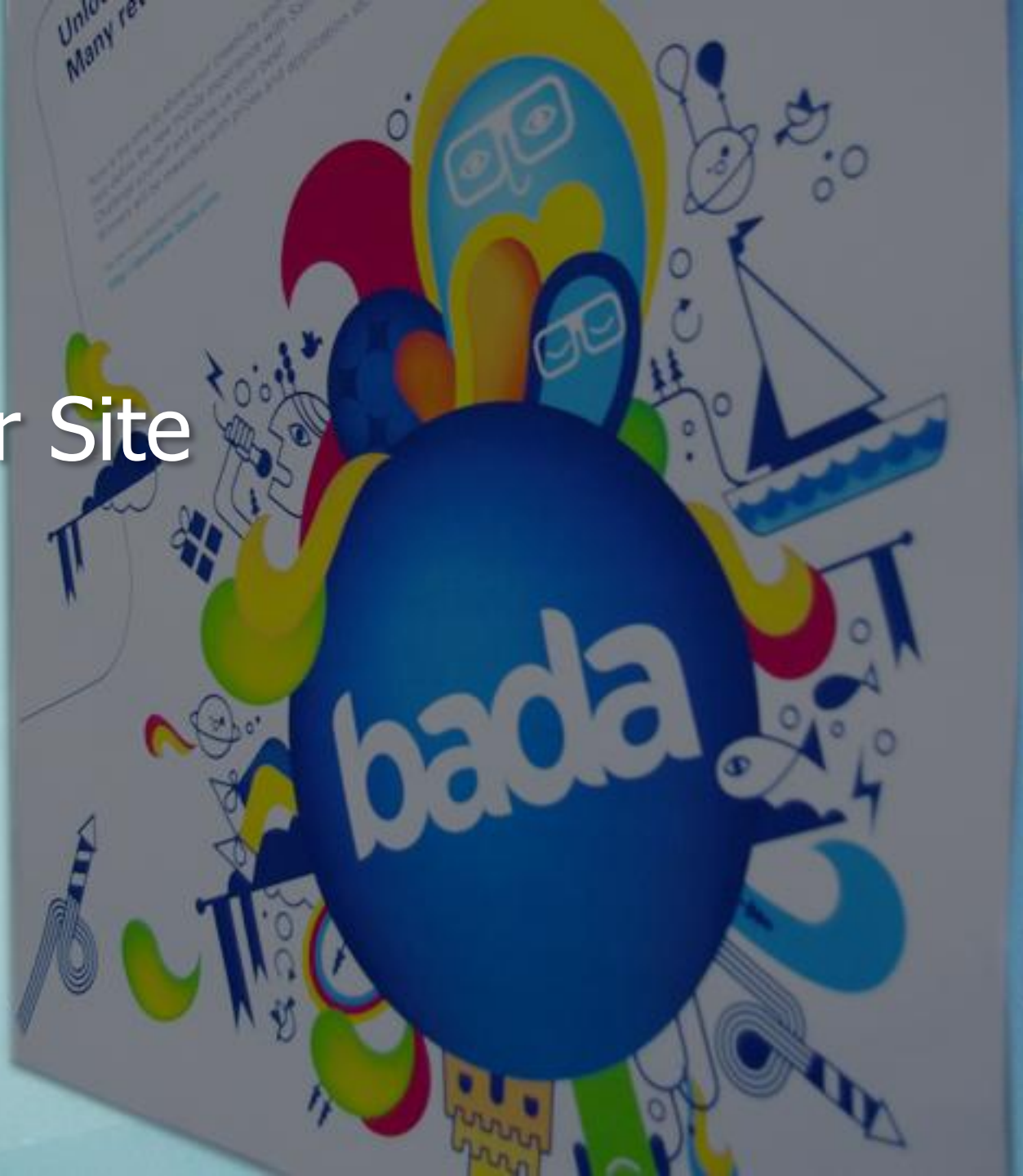
4 Samsung Apps Seller Office (seller.samsungapps.com)



- Verify and validate application
- Certify application
- Sell application
- Check sales statistics
- Remove application

Product Unregistration

Developer Site



Contents

- Overview
- Managing Your Applications
- Generating New Application Profiles
- Assigning Privilege Groups
- Setting up bada Components
- Selecting System Requirements
- Downloading the Manifest File



Overview (1/3)

bada Developers is a developer site that helps you develop and manage your application life-cycle. In it, you can:

- Generate new application profiles:
 - Name your application.
 - Set your application version number.
 - You can version your application. The default version is 1.0.0, but you can change it. The supported version range is from 0.0.1 to 35.35.1295.
 - You can change an application version during development with a higher version number.
 - Get your unique application ID.
- Assign privilege groups:
 - Assign privilege groups to be used in your applications.
 - Privilege group information is included in your application's `manifest.xml`.
- Set up bada components:
 - You can configure bada components that you selected while assigning privilege groups.
 - The Social, Location and Content components are configurable.



Overview (2/3)

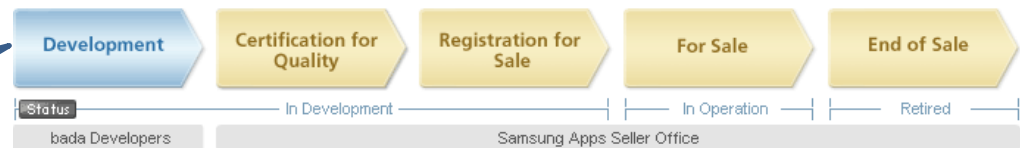
- Select system requirements:
 - Select minimum device requirements.
 - Selected application features are included in your application's `manifest.xml`.
 - Select CPU type, 3D accelerator version, screen size, and other hardware or software specifics to limit device model searches.
- Download `manifest.xml`:
 - Download `manifest.xml` for use in Eclipse IDE.
 - The `manifest.xml` file contains the application ID, privileges, application features, and other design and runtime information.



Managing Your Applications

The Way to Sell Your Application

The Application Manager allows you to manage the life-cycle of your mobile application with 5 steps as shown below.



There are 5 steps to sell your application:
Step 1. Generate your application profile.
Step 2. Request certification for your application.
Step 3. Submit your application to the Seller site.
Step 4. Now, your application is for sale.
Step 5. End of sale.

You can generate and develop your application on the bada developer site as the first step. Perform the other steps at the Seller site.

▪ Step1. Generate your application profile

Before starting to develop your application, you need some initial preparation, for example, filling in basic information and setting system requirements. You can get your unique application ID and manifest.xml file at this step.

➔ [Generate a New Application Profile](#)

* The following steps happen at the Seller Office:

To sell a bada application, refer to the "Publisher Guideline" from the Samsung Apps Seller Office and comply to its requisites. You can download the document after logging in at Samsung Apps Seller Office.

➔ [Refer Publish Guideline](#)

▪ Step2. Request a certification for your application

When your application is ready to be deployed, it needs to be packaged and uploaded to bada Developers for certification.

▪ Step3. Register your application at the Seller Office

To sell your application, you need to register your application to the Seller site. It will go on sale once it has been approved.

▪ Step4. Start your application sales

While your application is available for sale, you can't edit your application profile and component setup.

➔ [Go to Samsung Apps Seller Office](#)

▪ Step5. End your application sales

Once your application is retired, all resources and services will no longer be provided except for the status view service.

Generating New Application Profiles (1/2)

Application Manager

Generate Application

Select the type of Application you want to create.

Application Setup

New
 Update

Select application type.

You may enter a version number from 0.0.1 to 35.35.1295 only.

Application Version: 1 . 0 . 0

Enter application version.

ex) 35.35.1295
X : Major feature upgrade (0 ~ 35)
Y : Minor feature upgrade (0 ~ 35)
ZZ : build number or patch number (0 ~ 1295)

Start > Cancel >

Generating New Application Profiles (2/2)

Application Manager

New Application

01 | Create Application Profile

02 | Select Privileged API Groups

03 | Setup Components

04 | Setup System Requirements

05 | Review

Enter application name.

* Marked fields are Mandatory

▶ Create Application Profile

Application Name*

• Your application name must be between 4 and 20 alphanumeric characters in length. The only symbols allowed are the dash ("-") and underscore ("_").

Description

0/4000bytes

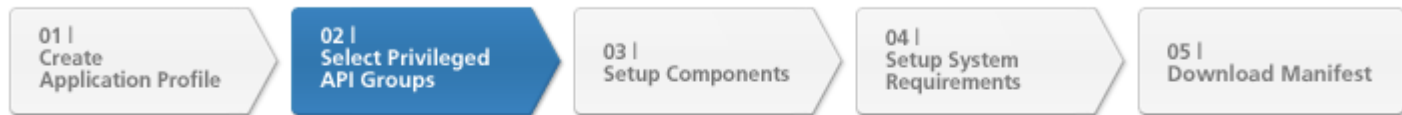
• The plaintext description of your application

For basic configuration settings and life-cycle management, each application needs a unique identifier. The application ID is created when you generate an application profile.

Assigning Privilege Groups

Application Manager

Application Wizard



▶ Select Privileged API Group

API Groups that can be selected is provided according to your membership level(Basic/Partner). The selected privileged API Group will undergo an authentication process later on so select only the ones you really need.

▸ Privileged API Group

Osp:: App ?	Select All	Select Clear
<input type="checkbox"/> NOTIFICATION		
Osp:: Content ?	Select All	Select Clear
<input type="checkbox"/> LOCAL_CONTENT		
Osp:: Locations ?	Select All	Select Clear
<input type="checkbox"/> LANDMARK	<input type="checkbox"/> LOCATION	<input type="checkbox"/> LOCATION_SERVICE

Select privilege groups.

Assign privilege groups to your application to include specific feature sets. This is similar to how you declare which namespaces your application uses.

Setting up bada Components

Application Manager

Application Wizard



▶ Setup components

Add and configure different bada components for your application. Click the name of a component to configure it in a new window.



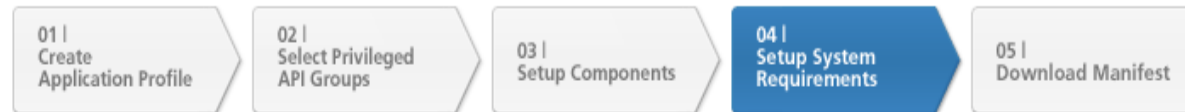
Click the component to configure.

Add and configure different bada components for your application. Click the name of a component to configure it in a new window.

Selecting System Requirements

Application Manager

Application Wizard



Define the system requirements by selecting the features that your application requires (to see the devices that have the features you want) or by selecting the device you want to develop an application for.

Find your bada Device

Find bada target devices for your application. You can search the bada devices that have some specific system features you want.

Search for Device Feature set

API Version: 1.0

CPU: ARM@CortexTM-A8

3D Accelerator: OpenGL@ES1.1, OpenGL@ES2.0

Screen Size: WWGA

Connectivity: Bluetooth, Wi-Fi

Sensor: GPS, Wi-Fi and cell-based positioning, Magnetic

Wave(S8500)

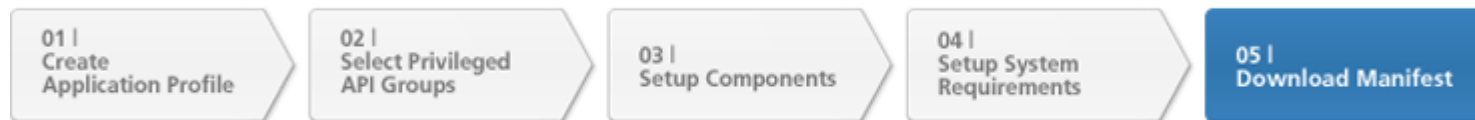
Select hardware or software features needed by your application.

Select the targeted device needed by your application.

Downloading the Manifest File

Application Manager

Application Wizard



Download Manifest.xml

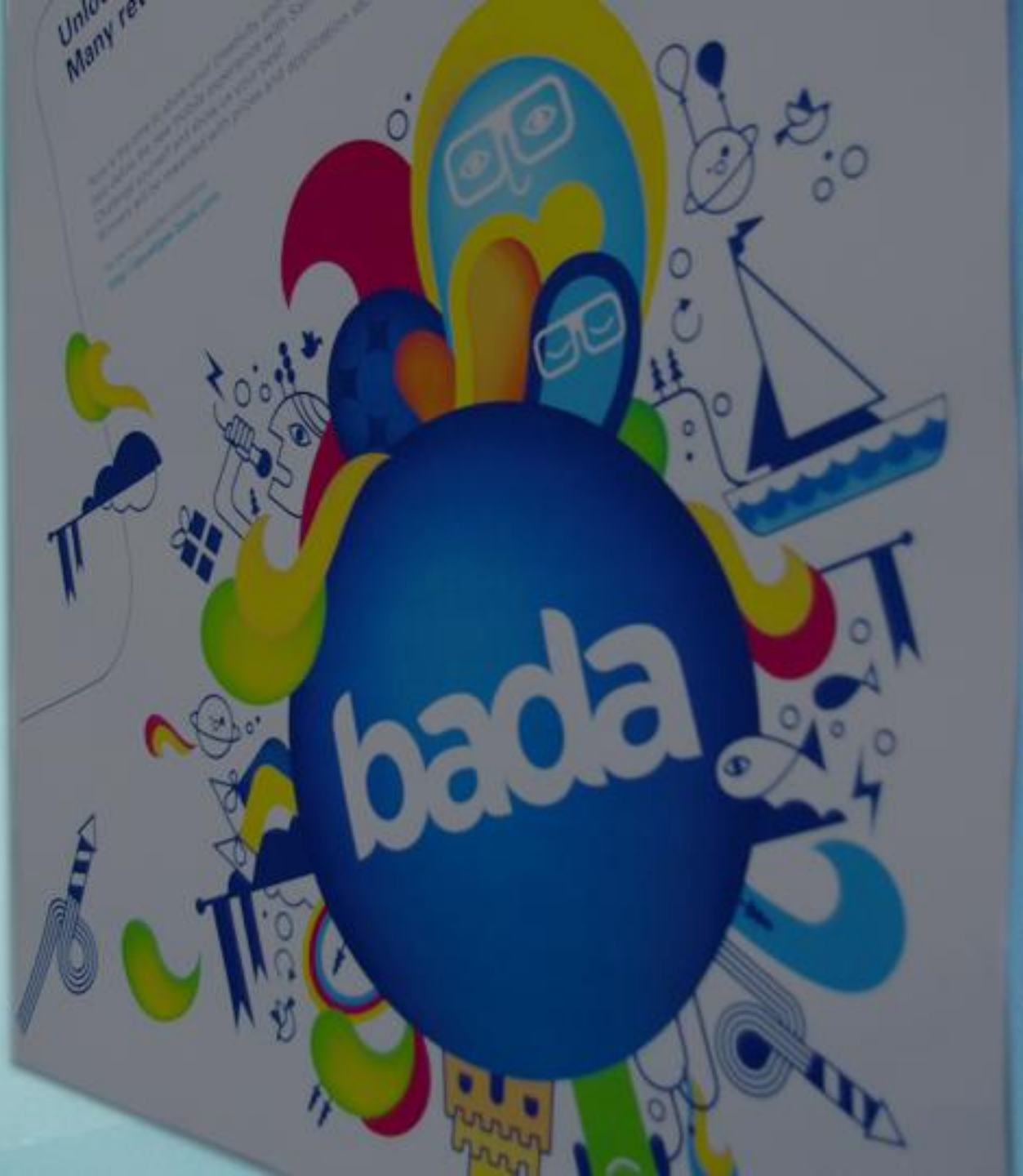
Copy manifest.xml file into root directory of your project and overwrite it.

Download 

Download your manifest file.

All applications require a manifest file that contains information specific to the platform, hardware, software, and application, including the application ID. Download the `manifest.xml` file for your application here.

SDK



Contents (1/2)

- Overview
- SDK Prerequisites
- Installing the SDK
 - Installing New Plug-ins
 - Updating the bada SDK
- Starting the IDE
- Development Process with the IDE
 - Importing Sample Applications
 - Creating Applications
 - Coding Applications
 - Setting Application Properties
 - Building Applications
 - Checking API Violations
 - Checking Privilege Violations



Contents (2/2)

- Debugging Applications
 - Installing a Device Driver for Test Devices
 - Installing the Test Root Certificate
 - Setting the Target Device to Debug Mode
 - Checking Debug Messages
 - Getting Debug Information
 - Stopping the Debug Process
- Running Applications
 - Checking Output Messages
 - Exiting Applications
- Testing Applications
 - Manipulating Simulator
 - Testing Applications with Remote Test Lab
 - Error Codes
- Packaging Applications
 - Packaging Applications on Command Line
- Upgrading Applications
- IDE Text Editor
- Resource Monitor
- Output



Overview

- Prepare for application development:
 - Download the bada SDK from the bada developer site.
 - Install the SDK.
- bada SDK:
 - Platform binaries and libraries
 - Header files
 - Simulator
 - Documentation
 - Sample applications
 - bada IDE:
 - Eclipse CDT-based (C/C++ Development Tools)
 - Application Wizard
 - UI Builder / Resource Explorer
 - Resource Monitor
 - bada Testing Tool



SDK Prerequisites

To install and run the bada SDK, you need:

- Computer with :
 - Microsoft Windows® XP, Windows® Vista or Windows® 7
 - 1.4 GB RAM or higher
 - 1.8 GB free disk space or higher
 - Local administrator rights
 - Note that the Simulator screen size is 480 x 800. Screen resolutions under 800 do not show normally for OpenGL® applications. This is not a defect; this is due to the Windows operating system.
- Web browser to access the bada developer site.
The supported Web browsers include:
 - Internet Explorer 6.0 or higher
 - Firefox 3.0 or higher
 - Safari 3.2 or higher
- The bada membership to download the SDK from the bada developer site.



Installing the SDK

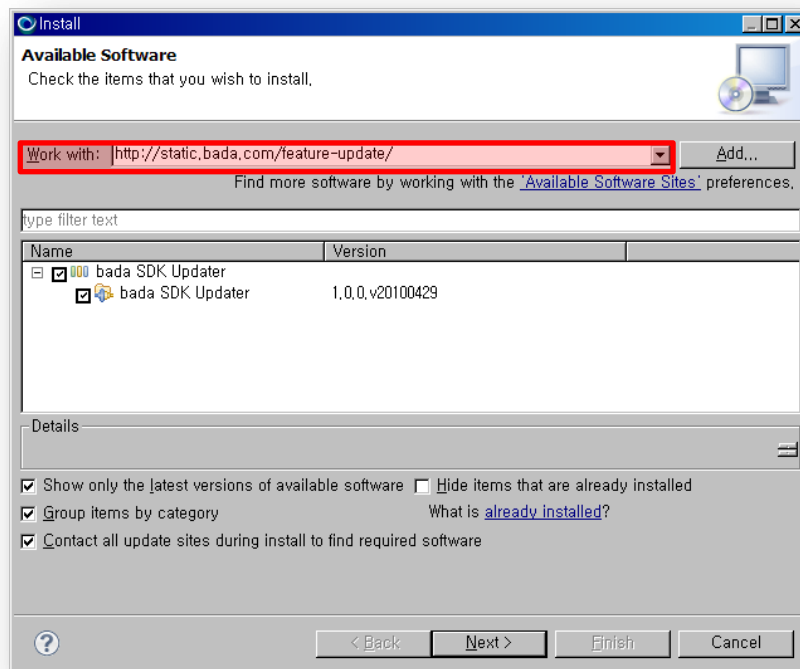
- The SDK uses a standard installation wizard. To install the SDK on your development machine:
 1. Download the SDK installer, `bada_SDK_<version>.exe`, from the [bada developer site](#).
 2. Run the SDK installer.
 2. Select the required language packs and click **Next**.
 3. Click **Install**.
- Once the SDK installer is finished, the SDK is installed and configured.
- If you need to add more language packs to your SDK later on, do not reinstall the SDK with the SDK installer, since it can cause problems for your development environment. Instead, download the `bada_SDK_<version>_<model>_LP#.exe` patch installation packs you need from the [bada developer site](#) and run the corresponding language pack installers.



Installing New Plug-ins

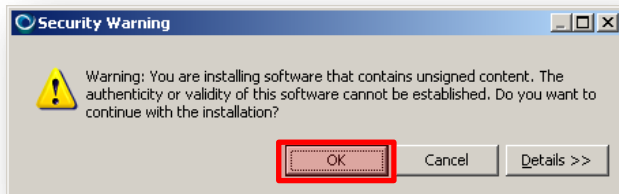
You can install new plug-ins for the bada IDE:

1. In the bada IDE, go to **Help > Install New Software**.
2. Select <http://static.bada.com/feature-update/> as the software site that you work with.
3. Select the plug-ins to install and click **Next** to continue and complete the installation process.

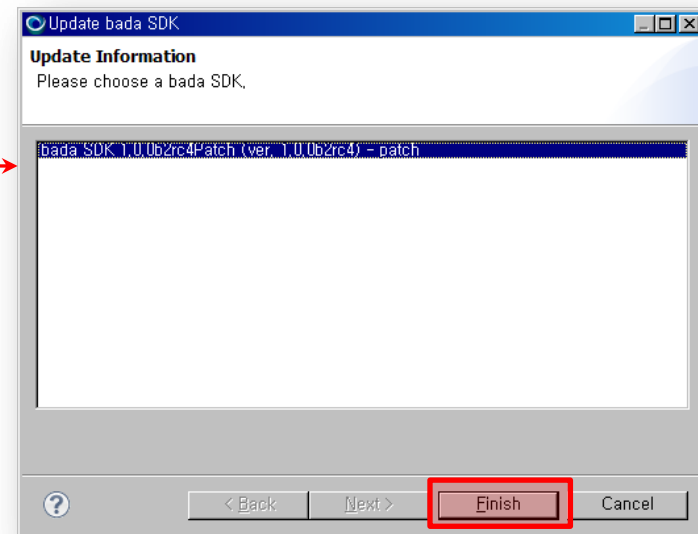
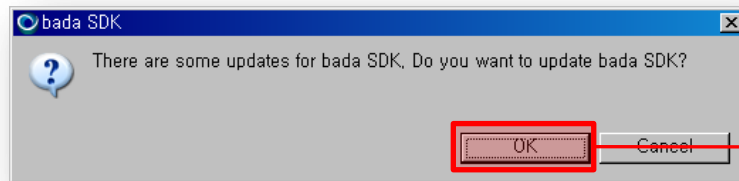


Updating the bada SDK

- Install the bada SDK Updater by going to **Help > Install New Software** in the bada IDE.
 - When a security warning dialog is displayed, click **OK** to ignore it.



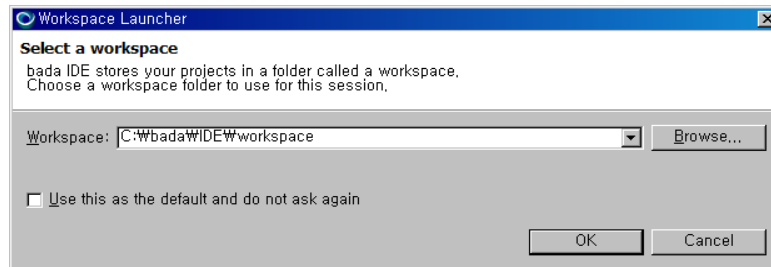
- With the bada SDK Updater installed, the bada IDE checks for SDK updates or new SDK versions each time it starts.



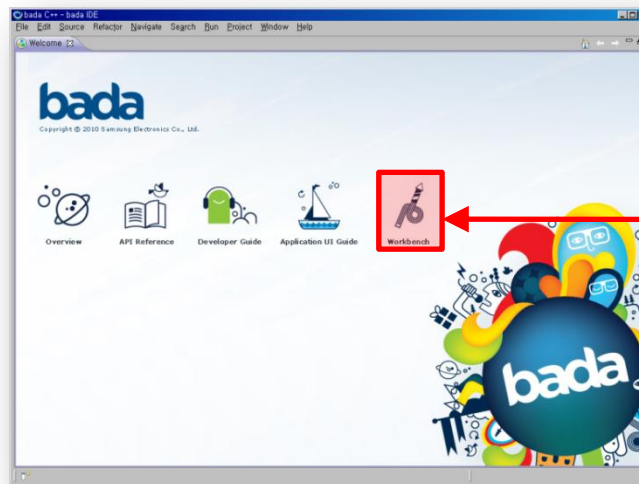
Starting the IDE

You can start developing a new bada application with bada IDE.

- Start bada IDE and specify a workspace folder. If you start with a new version of the IDE, do not use a workspace created for a previous IDE version.

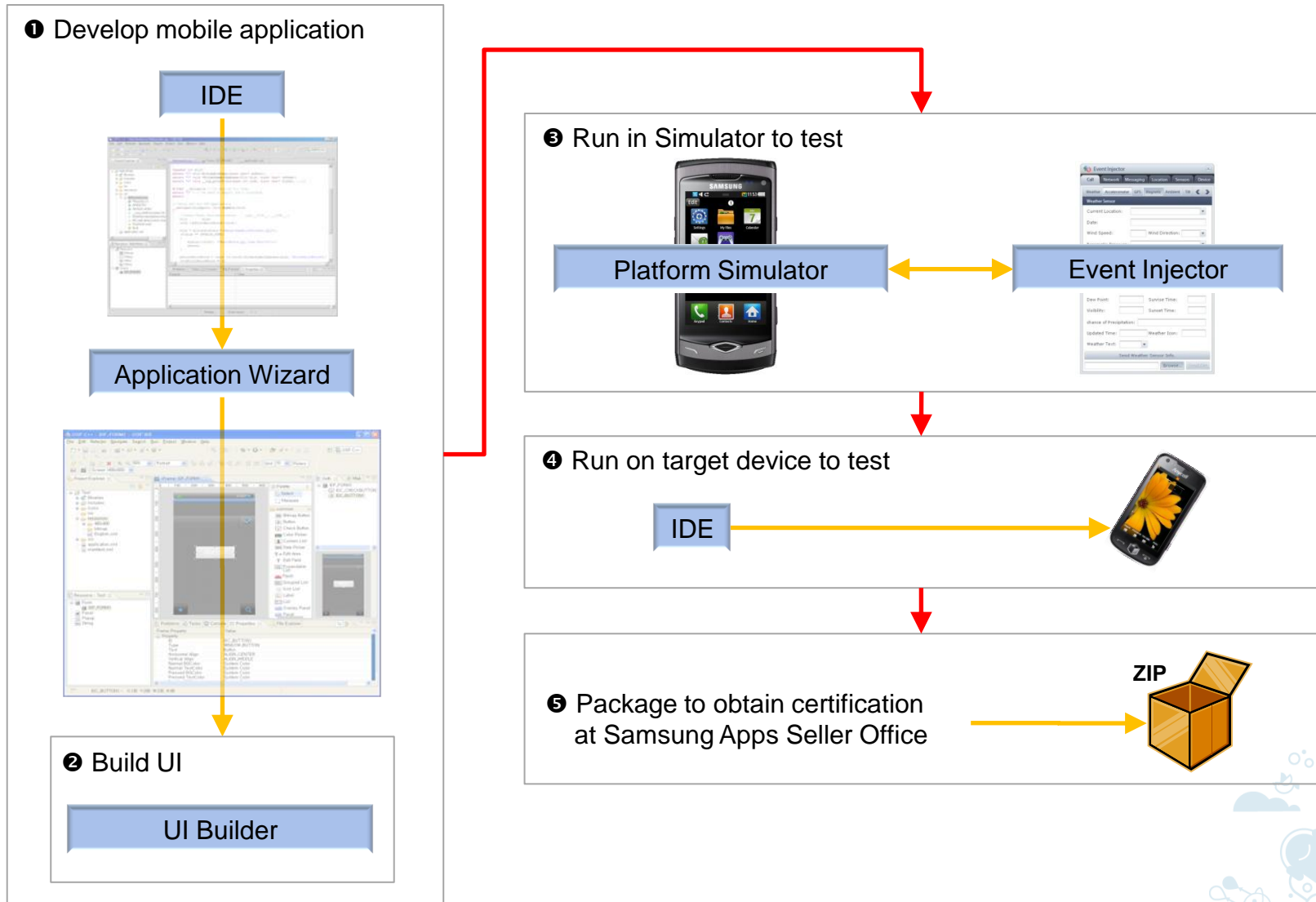


- When you start the IDE for the first time, a welcome page is displayed with more information on bada and on developing applications with the bada IDE.



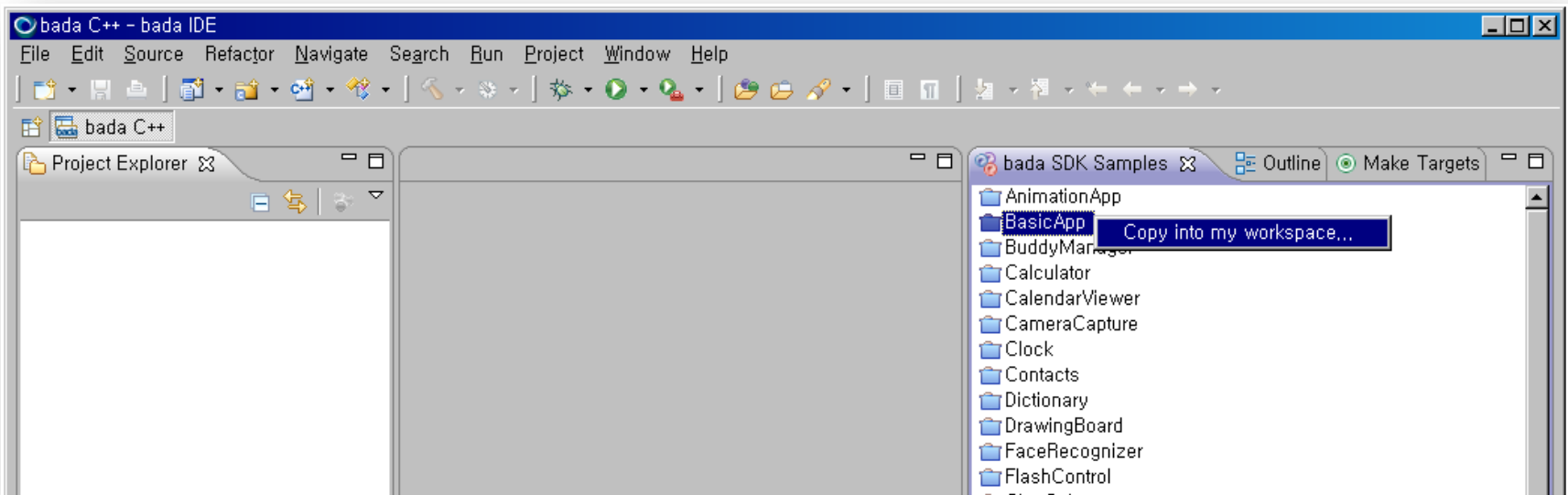
Click this icon to go to the development view

Development Process with the IDE



Importing Sample Applications (1/2)

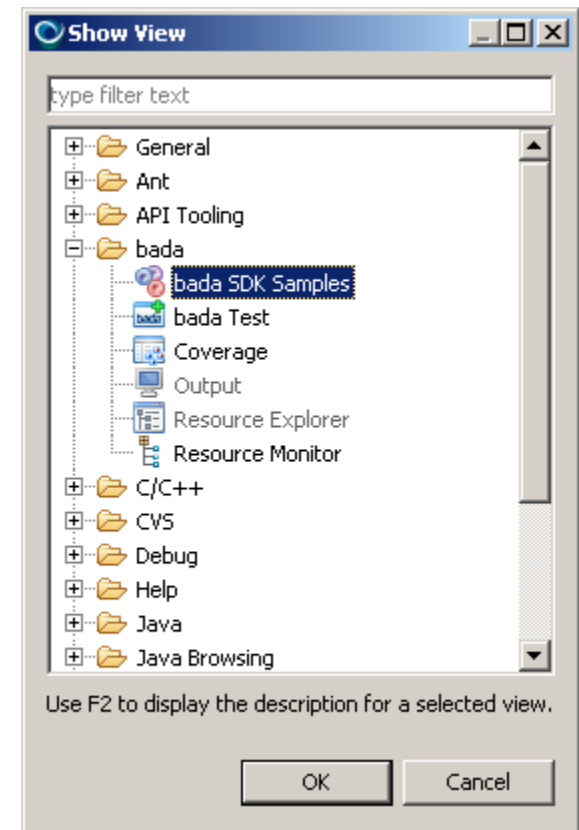
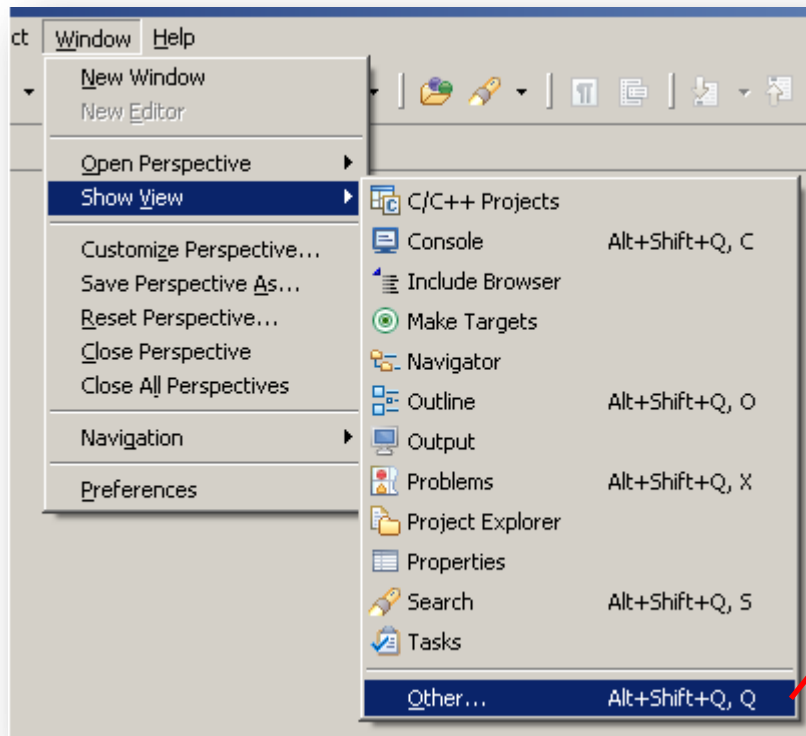
Right-click on sample to be imported and click **Copy into my workspace...** in **bada SDK Samples**.



Importing Sample Applications (2/2)

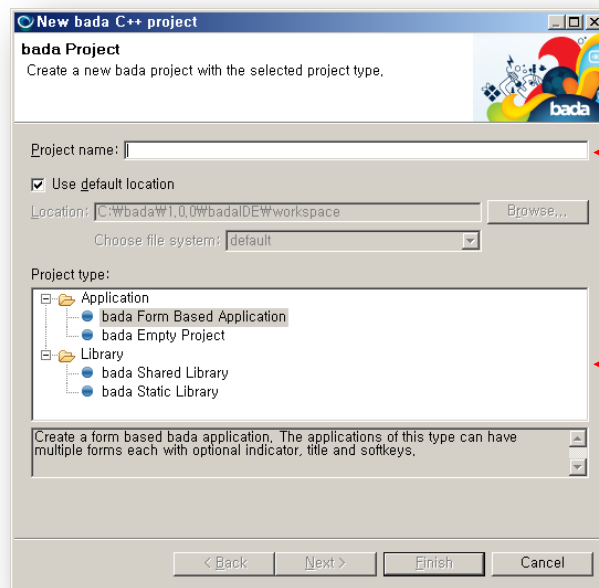
You can load bada SDK samples with the following steps:

1. Go to **Window > Show View > Other...**
2. Select **bada > bada SDK Samples**.



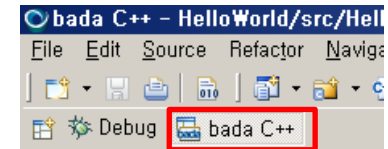
Creating Applications (1/7)

- You can create new bada applications with existing application templates. (Check if the current perspective is bada C++.)
- Go to **File > New > bada Application Project**.



Give a name to your project.

Choose a project type.



- New application projects automatically fill in basic code and let you configure the project for different build configurations and devices. The project creation process in the IDE consists of multiple windows. In each window, check and set settings, and click the **Next** button.

Creating Applications (2/7)

There are four bada project types.

1. bada Form Based Applications:

- This template is suitable for creating a simple project based on a form.
- This template contains the basic application functionality as well as the functionality for drawing a form on the device screen.

2. bada Empty Project:

- This template is suitable for creating a project with only project files without any source or include files. It is used for importing the existing source files.

3. bada Shared Library:

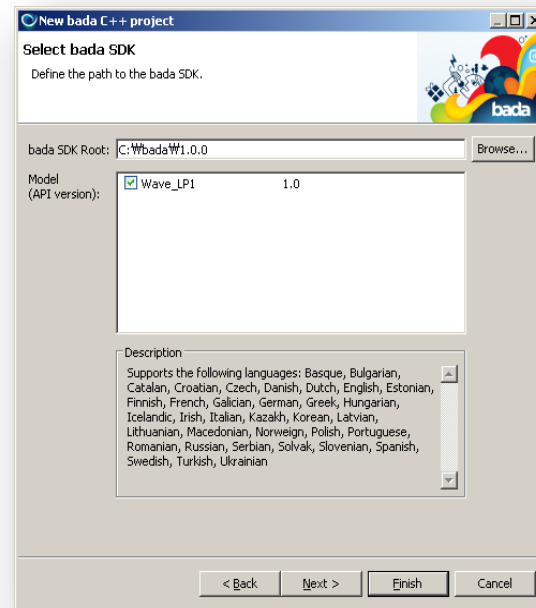
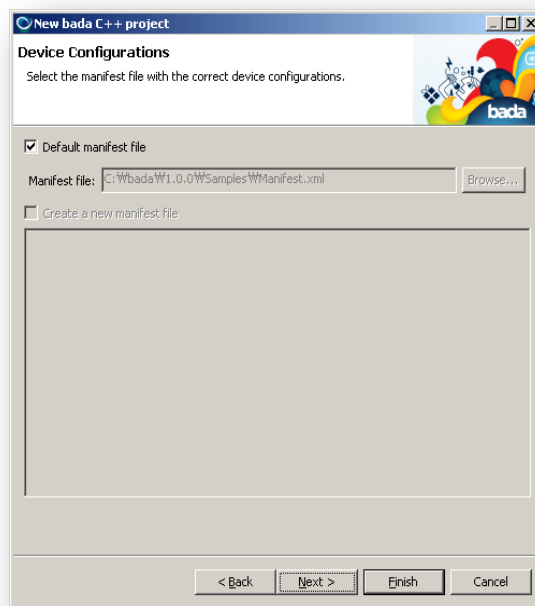
- This template is suitable for creating an application library with shared libraries. If you select this template, you must make sure that the Linker of the IDE can access the external libraries at the build time. Therefore, you must define the path to the libraries in the project settings. When you build the project, the IDE creates the links to the external shared libraries.

4. bada Static Library:

- This template is suitable for creating an application library with static libraries. If you select this template, you must make sure that the Linker of the IDE can access the external libraries at the build time. Therefore, you must define the path to the libraries in the project settings. When you build the project, the IDE creates the links to the external static libraries.

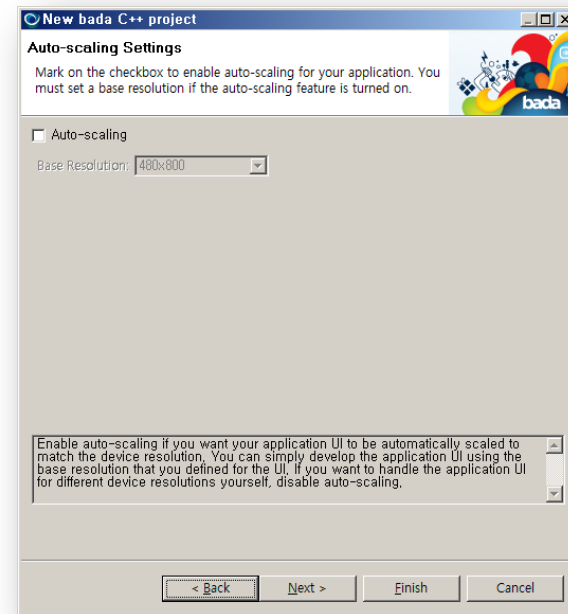
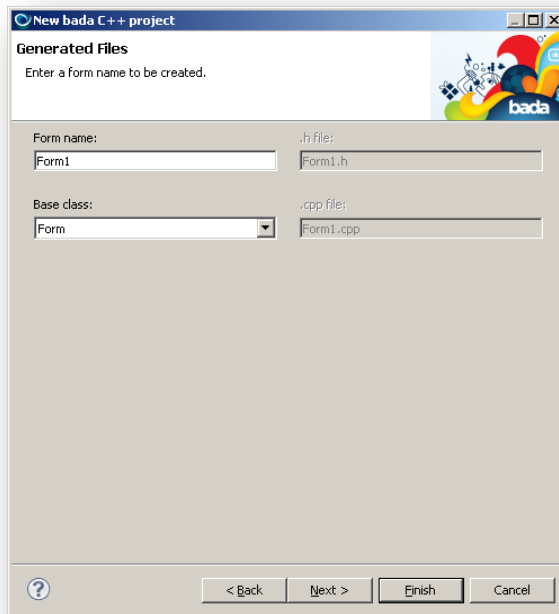
Creating Applications (3/7)

- On the Device Configurations page:
 - To use a default device configuration, select the **Default manifest file** check box. An application with the default configuration cannot be sold at Samsung Apps.
 - To use a new manifest file, first create and download a manifest file on the bada developer site. Then, on the **Device Configurations** page, clear the **Default manifest file** check box, and browse to the downloaded file.
- On the Select bada SDK page:
 - Check that the path to the bada SDK root is correct.
 - Check that the device model and the related API version for which you are developing your application are correct. The `manifest.xml` file contains the API version of the application, and you must select a model with the same API version as in the manifest file.



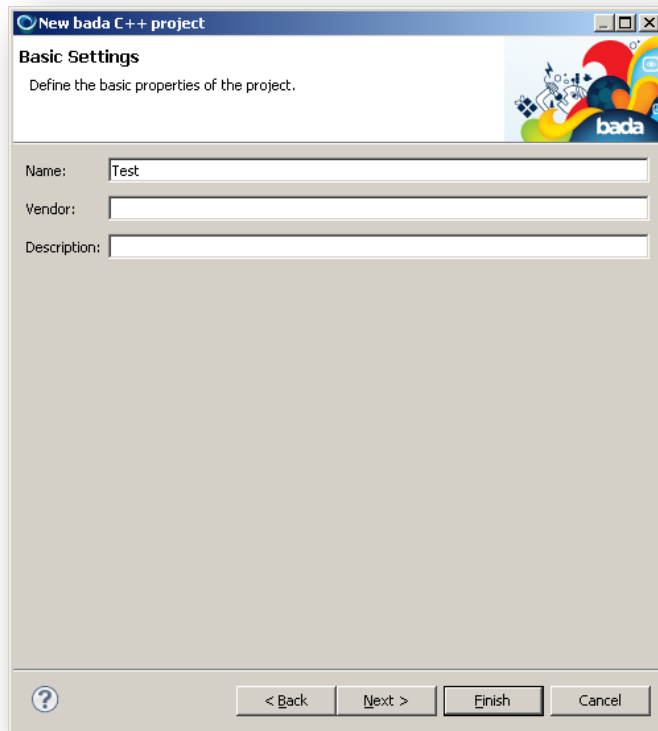
Creating Applications (4/7)

- On the Generated Files page:
 - This page is only shown for form based applications.
 - Enter the form name.
- On the Auto-scaling Settings page:
 - Select the **Auto-scaling** check box to enable your application UI to scale automatically and match the device resolution.
 - If you enable auto-scaling, select a base resolution from the **Base Resolution** drop-down list.



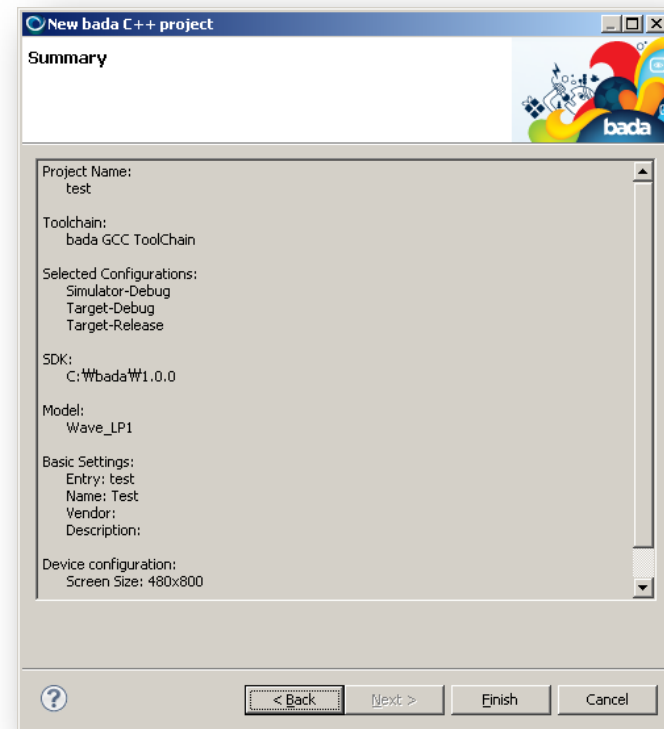
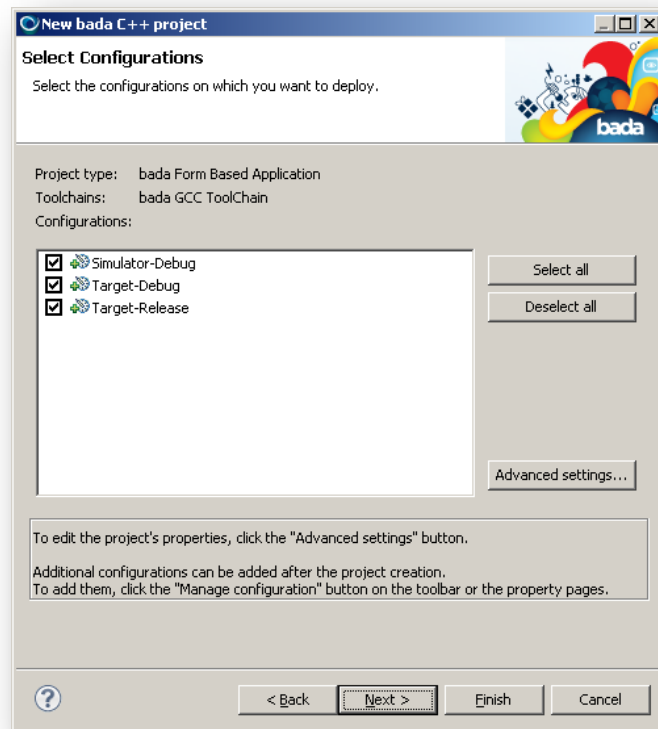
Creating Applications (5/7)

- On the Basic Settings page:
 - Enter a description for your application.



Creating Applications (6/7)

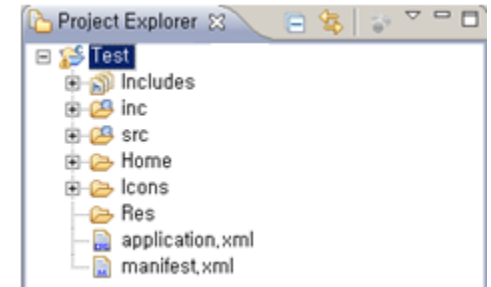
- On the Select Configurations page:
 - Select the device configurations for the application deployment scenarios.
- On the Summary page:
 - Verify the settings for your new application and click the **Finish** button.



Creating Applications (7/7)

Project folder structure

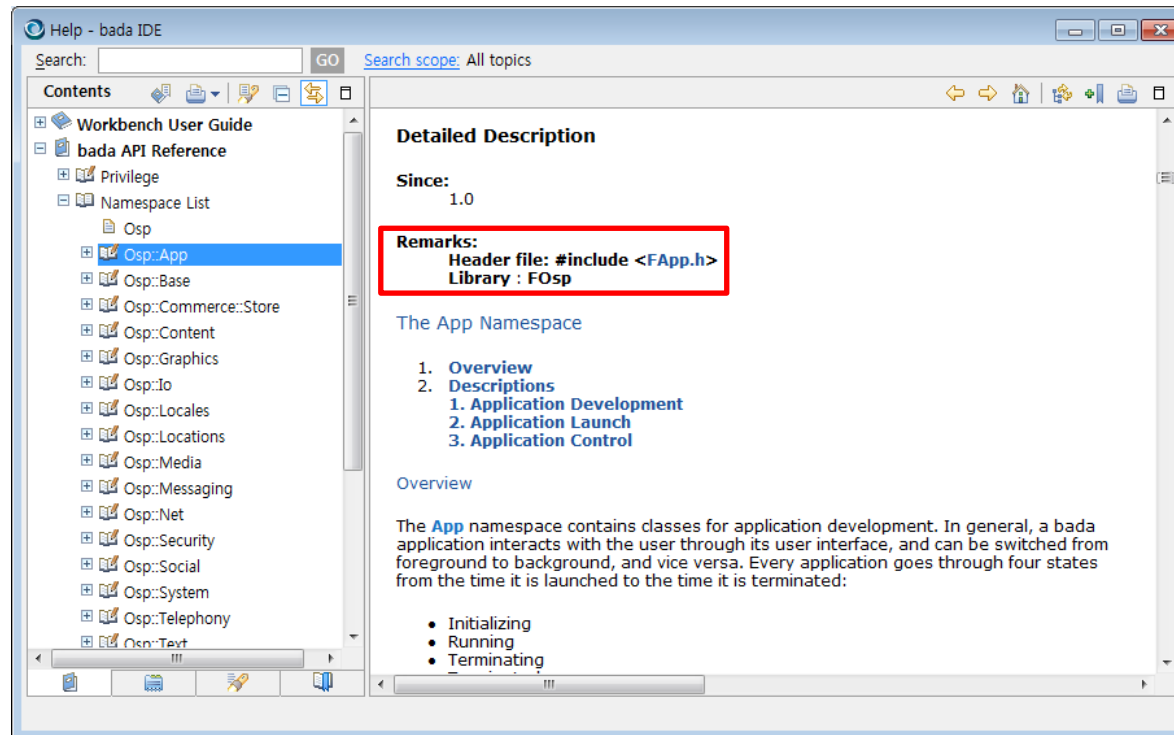
- `/Includes` refers to the bada platform include files.
- `/inc` has include files created by the application wizard.
- `/src` has source files created by the application wizard.
- `/Home`
 - Readable and writable files and folders for your application are copied to this folder.
 - The 'Share' folder name cannot be created in `/Home` because it is a reserved folder.
 - It is accessed by `/Home` in source code.
- `/Res`
 - Read-only files and folders for your application are copied to this folder.
 - UI Builder files are created in this folder.
 - It is accessed by `/Res` in source code.
- Although files and folders with a '.' prefix are copied to `/src`, `/inc`, `/Home` and `/Res`, they are ignored and are not copied to the target device.



Coding Applications (1/3)

Add include files:

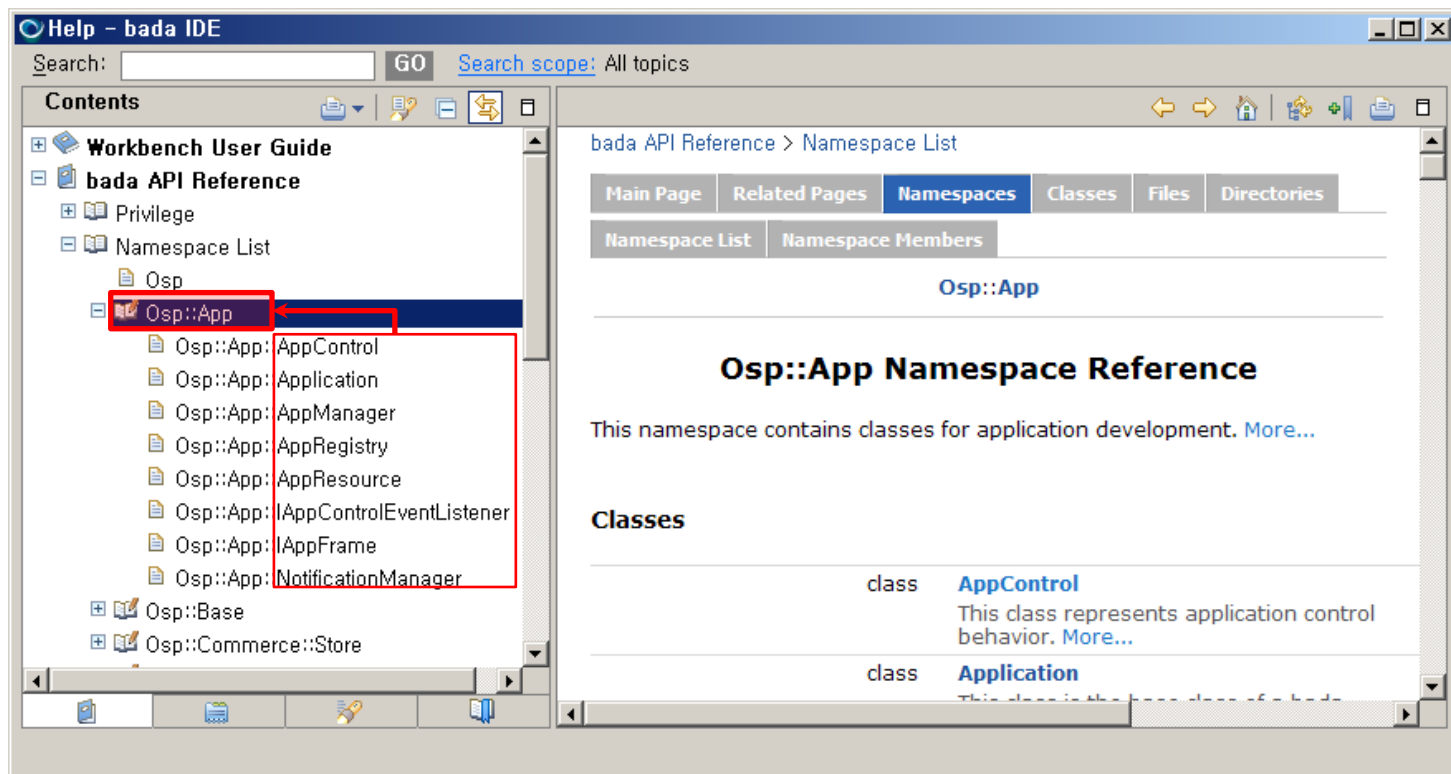
- Required include files are listed under **Remarks** in the bada API Reference.
- For example, if your application uses classes in `Osp::App::AppControl`, you must include `FApp.h` as shown under **Remarks** on the `Osp::App` page.



Coding Applications (2/3)

Add namespaces:

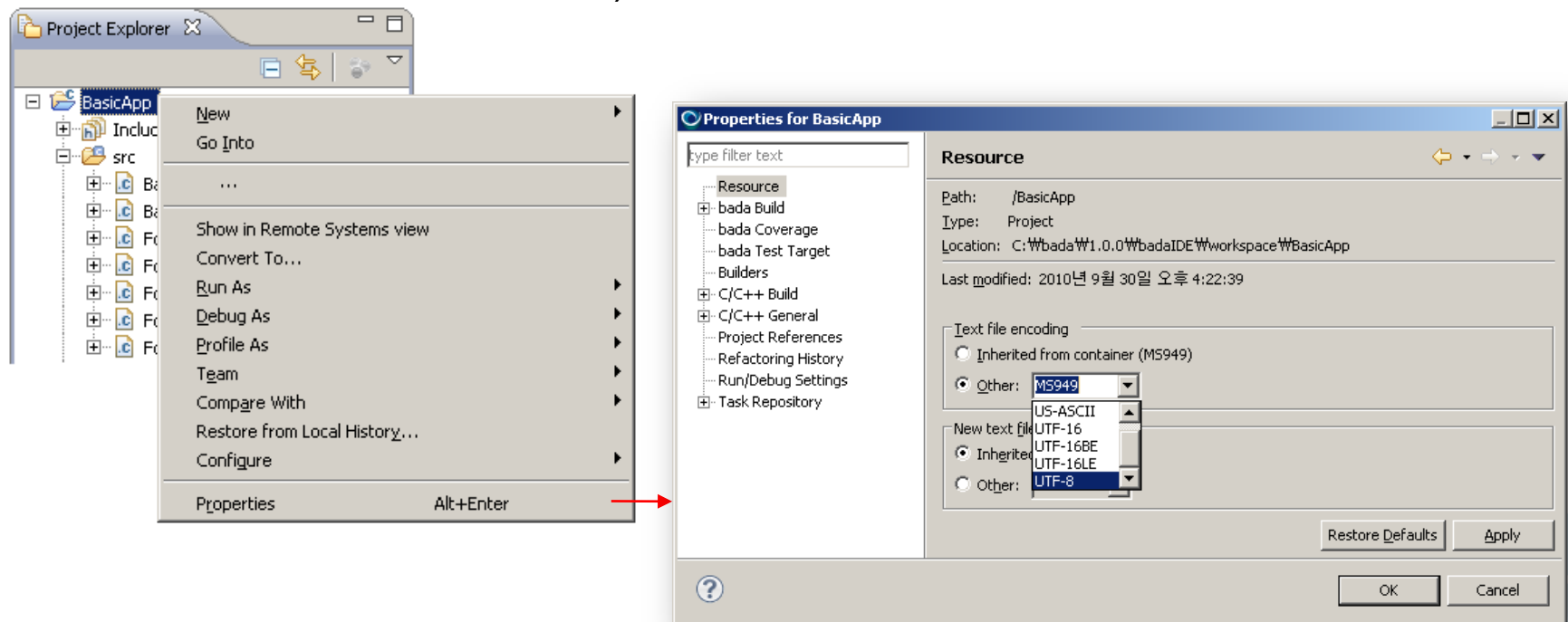
- Namespaces are listed in the bada API Reference.
- For example, if your application uses classes in the `Osp::App` namespace, you must add `using namespace Osp::App` in your CPP files.



Coding Applications (3/3)

Change the encoding format for your source files if the source files include UNICODE text:

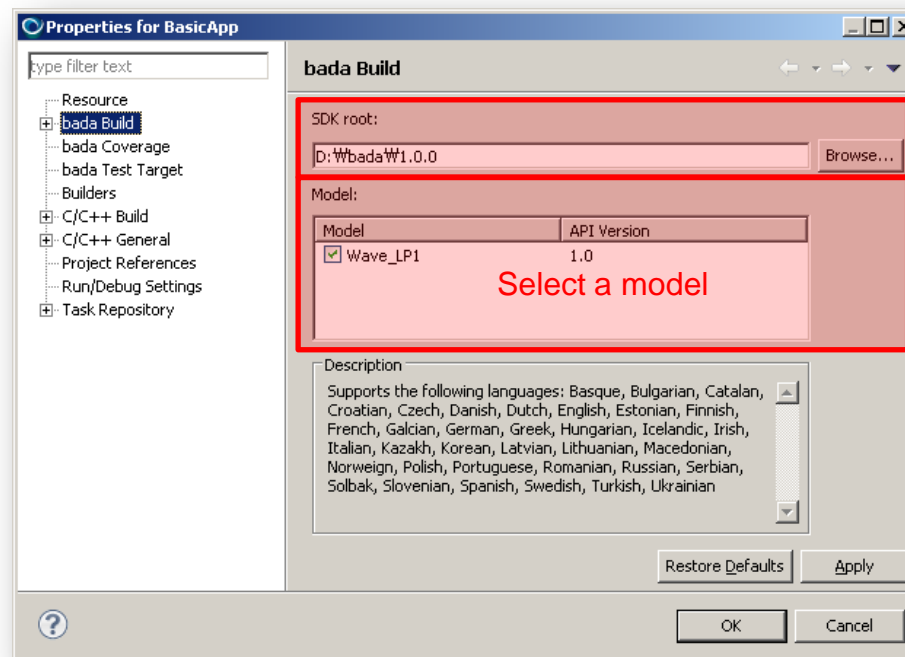
- Right-click the project and select **Properties > Resource**.
- Select an applicable value from the **Other** drop-down list (for example, **UTF-8** in case of Korean).



Setting Application Properties (1/12)

You can select an SDK root and a model:

- Select an SDK path among installed SDKs.
 - The latest installed SDK is the default SDK root.
- Select a destination model for developing your application in the selected SDK root.
 - The `manifest.xml` file contains the API version of the application, and you must select a device model with the same API version as in the manifest file.



Setting Application Properties (2/12)

You can set application icons:

- To add icons to an application, right-click the project and select **Properties > bada Build > Application Information**.

Application Information

Entry: BasicApp

Name	BasicApp	New	Delete
English			

Vendor:

Description:

Icons	Type1	Type2	Browse...
Main Menu:	BasicApp.png		Browse...
Setting:			Browse...
Ticker:			Browse...
Quick Panel:			Browse...
Launch Image:			Browse...

AutoScaling

Auto-scaling

Base Resolution: 480x800

Restore Defaults Apply OK Cancel

Make sure the entry name is identical to the name of the executable binary.

Do not remove the "English" title. It is necessary.

Change the default images to images that suit your application.

Set the auto-scaling configuration.

Set application names corresponding to each language

Set icons

Set auto-scaling

Setting Application Properties (3/12)

Add the icons based on the descriptions given in the table below:



Icons	Mandatory	Format	Size ¹	Description
MainMenu (A) ²	Yes	32-bit PNG with alpha	Type 1: 100 x 96 Type 2: 50 x 47	The image is displayed on the main menu.
Setting	-	-	-	Reserved
Ticker (B) ³	No	32-bit PNG with alpha	Type 1: 32 x 32 Type 2: 18 x 18	The image is displayed on the indicator area when the application receives an event.
QuickPanel (C)	No	32-bit PNG with alpha	Type 1: 68 x 74 Type 2: 34 x 37	The image is displayed on the quick panel when the application receives an event.
LaunchImage (D)	Yes	JPEG or PNG	Type 1: 480 x 800 Type 2: 240 x 400	The image is displayed during application initialization.

1 Type 1 refers to images for the WVGA resolution, while type 2 refers to images for the WQVGA resolution.

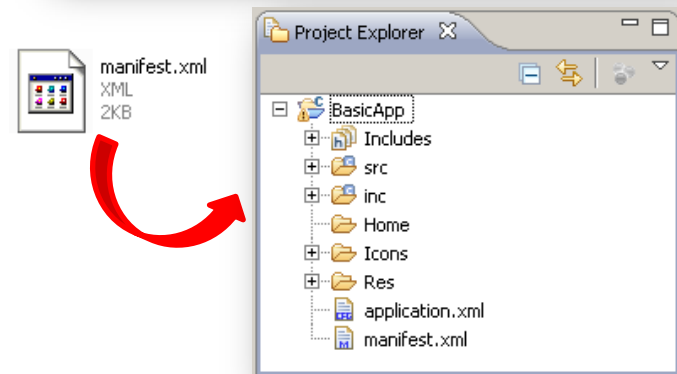
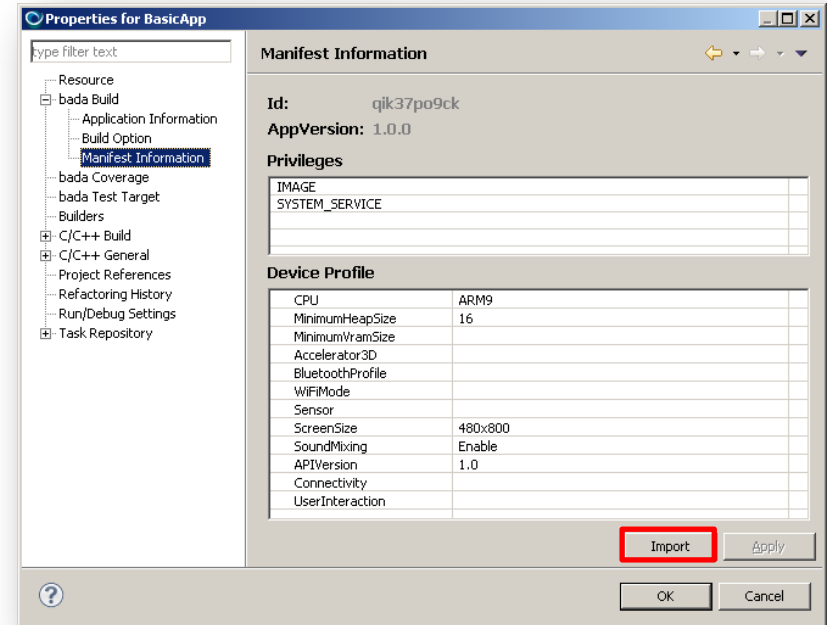
2 To display the icon appropriately, maintain a 1-pixel margin on all four sides of the icon.

3 To display the icon appropriately, maintain a 1-pixel margin at the top of the icon.

Setting Application Properties (4/12)

You can change the manifest file:

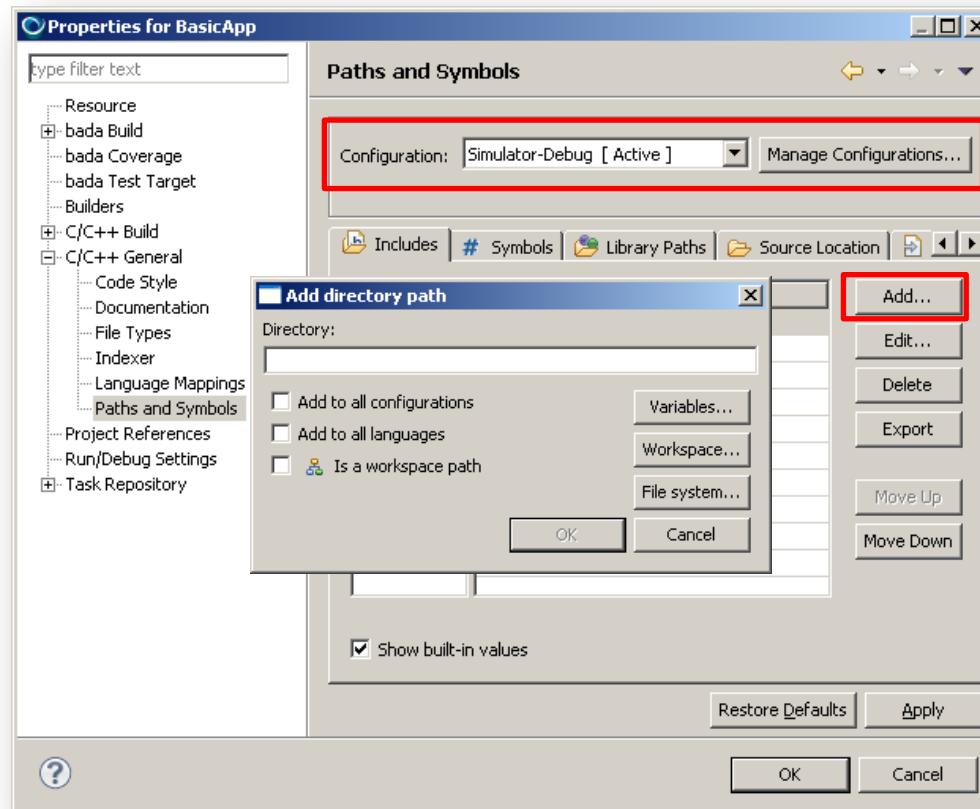
- To change the manifest file:
 - Right-click the project and select **Properties > bada Build > Manifest Information**.
 - Click **Import** and select the file to be used.
- You can also change the manifest file by simply dragging and dropping a new file onto the root project directory in the **Project Explorer** view.
- If you change the manifest file, you must rebuild the application.



Setting Application Properties (5/12)

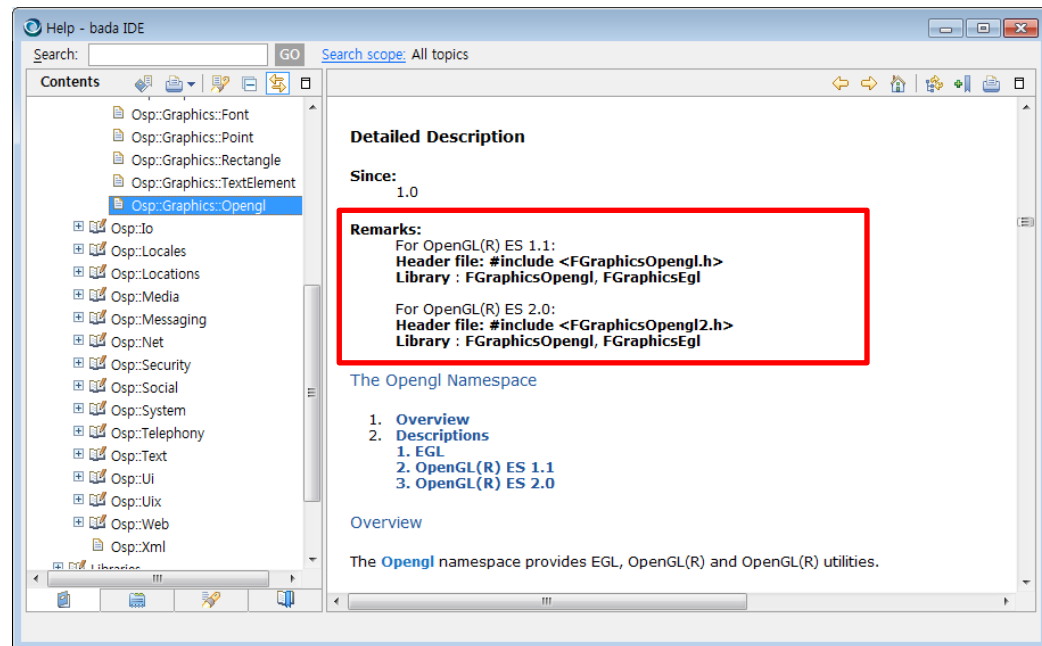
You can add include paths for your application:

1. Right-click the project and select **Properties > C/C++ General > Paths and Symbols**.
2. Select a configuration and add include paths.



Setting Application Properties (6/12)

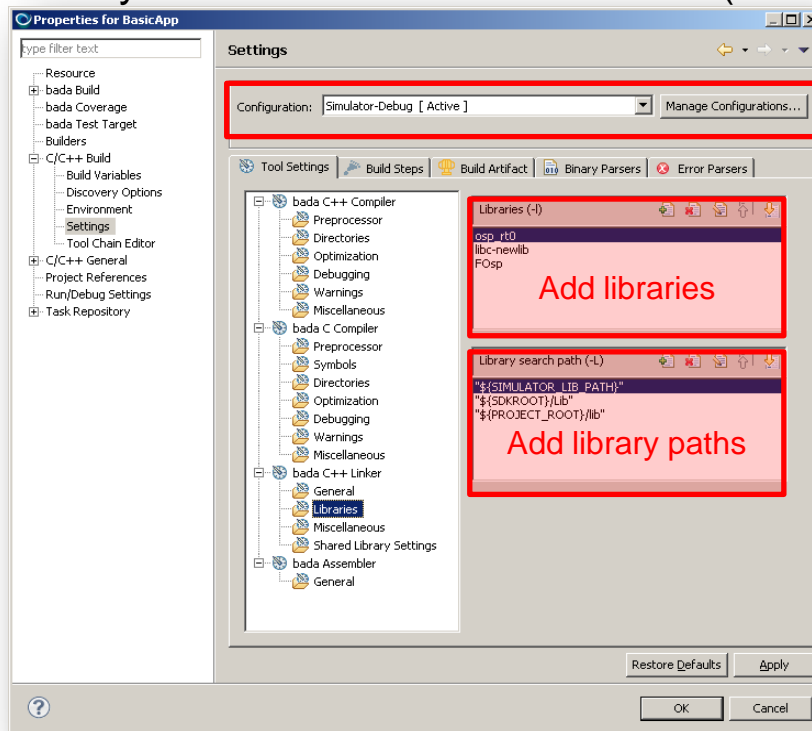
- You can add bada libraries, external libraries, and library search paths for your application.
- The default bada library, FOsp, is added automatically by the application wizard, but you need to add the other required libraries manually.
 - For example, if your application uses OpenGL® ES, you must add the required libraries, shown in the bada API Reference under **Remarks**, to the library list.



Setting Application Properties (7/12)

To add library search paths and libraries for the Simulator-Debug configuration:

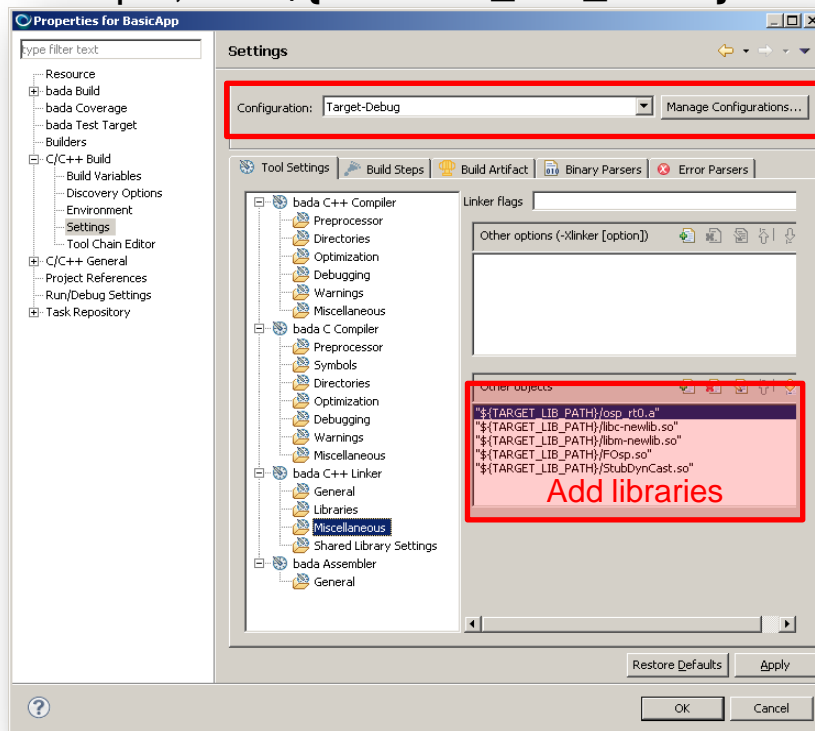
1. Right-click the project and select **Properties > C/C++ Build > Settings**.
2. Select the **Simulator-Debug** configuration and add library search paths and libraries.
 - Add library names without file extensions (for example, **FGraphicsOpengl**).



Setting Application Properties (8/12)

To add static libraries for the Target-Debug or Target-Release configuration:

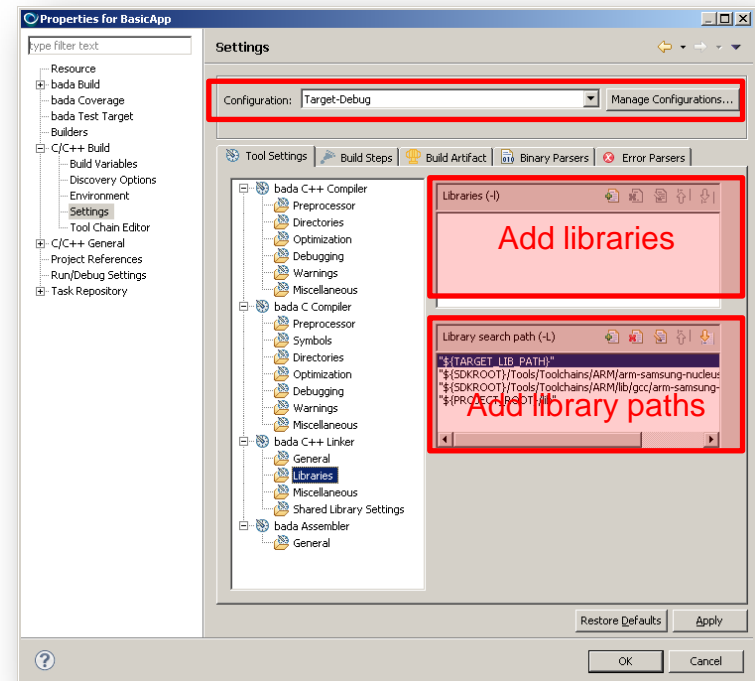
1. Right-click the project and select **Properties > C/C++ Build > Settings > bada C++ Linker > Miscellaneous**.
2. Select a configuration and add libraries.
 - For example, add `${TARGET_LIB_PATH}/YOUR_LIBRARY.a`



Setting Application Properties (9/12)

To add shared libraries for the Target-Debug or Target-Release configuration:

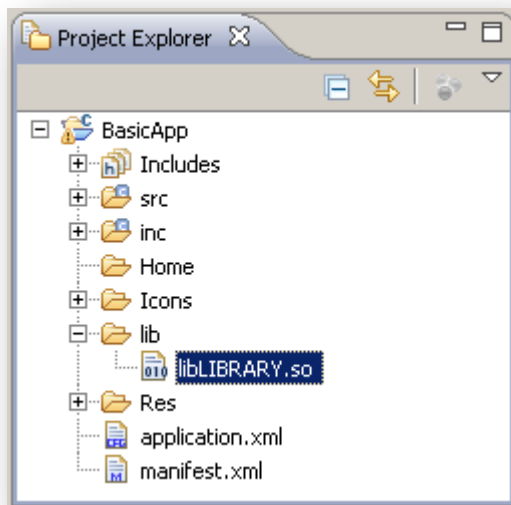
1. Right-click the project and select **Properties > C/C++ Build > Settings**.
2. Select the **Target-Debug** or **Target-Release** configuration and add library search paths and shared libraries.
 - The file name of shared libraries must start with “lib” (for example, `libLIBRARY.so`).
 - Add library names without prefix and file extensions (for example, `LIBRARY`).



Setting Application Properties (10/12)

To add shared libraries to the file system of the Simulator or the target device:

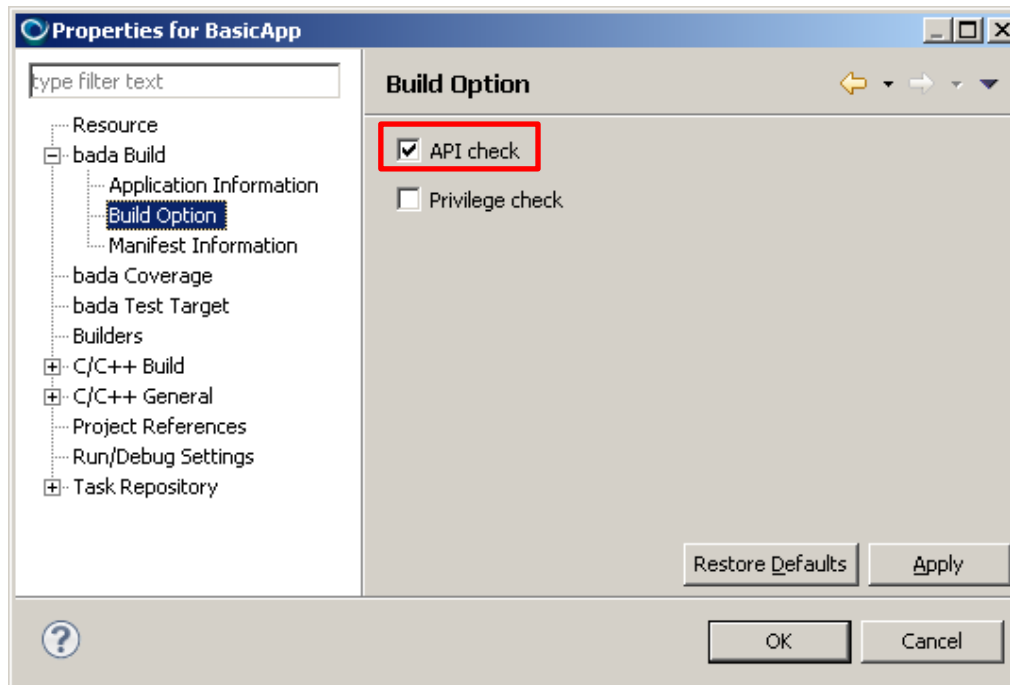
1. Create a `lib` folder in your project folder.
2. Copy the shared libraries to the newly created `lib` folder.
To use the shared libraries in the Simulator, also copy them to the `\<BADA_SDK_HOME>\Model\<Device_model>_LP#\Simulator` folder.



3. Add the library search path and the shared libraries at **Properties > C/C++ Build > Settings > bada C++ Linker > Libraries**.

Setting Application Properties (11/12)

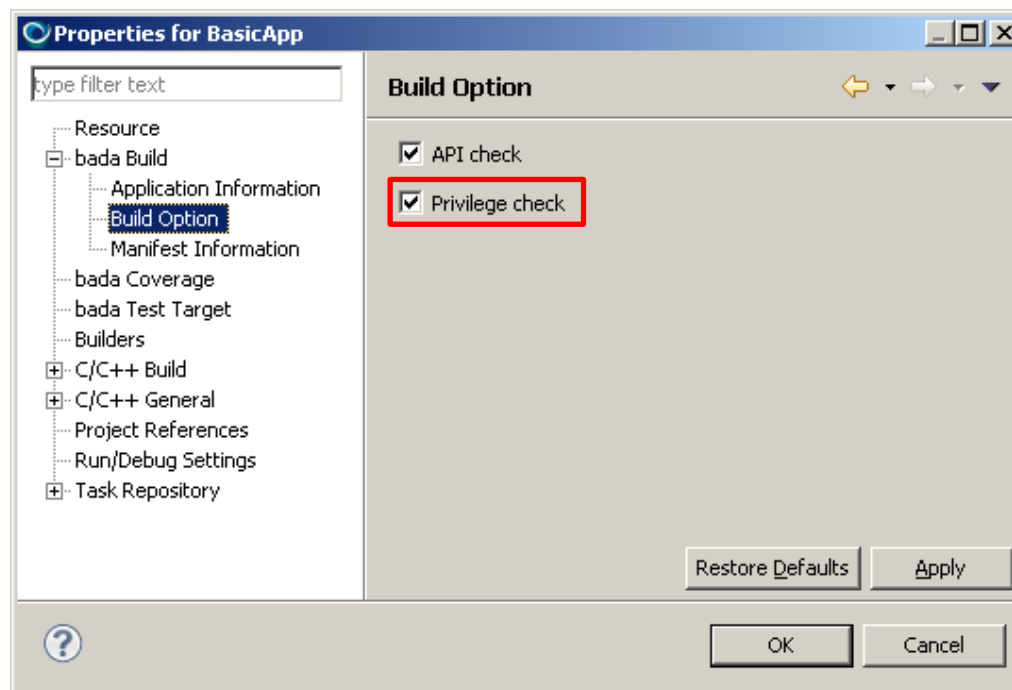
- The versions of API used in the manifest file and your application must match. To enable the version match check:
 1. Right-click the project and select **Properties > bada Build > Build Option**.
 2. Select the **API check** check box.



- The check is automatically performed when you build the application. If the application has conflicting API versions, the build fails and the results are displayed in the **Problems** view.

Setting Application Properties (12/12)

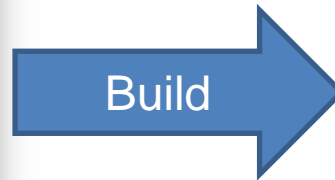
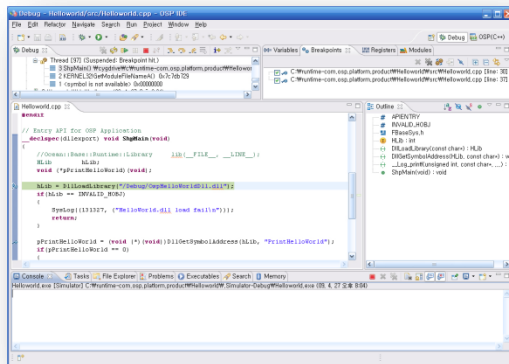
- You can check the source code in your project for any violation of privileges. To enable the privilege check:
 - Right-click the project and select **Properties > bada Build > Build Option**.
 - Select the **Privilege check** check box.



- The check is automatically performed when you build the application, and the results are displayed in the **Problems** view.

Building Applications (1/2)

Building applications is essentially the same as for non-mobile applications. Samsung bada adds in an additional Simulator build for testing on your local machine, speeding up your development time.



Run in Simulator or on Device



Building Applications (2/2)

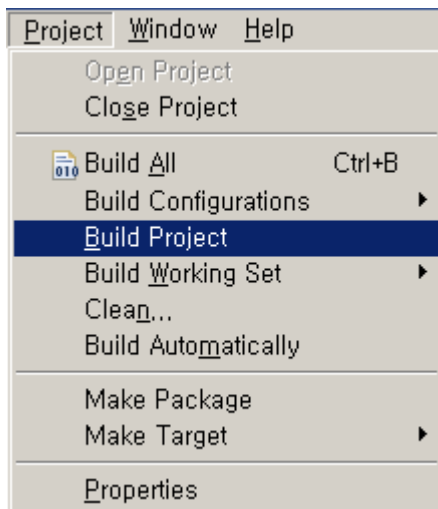
To build your application:

1. Select a build configuration.

There are separate configurations for debugging and testing the application on the Simulator or the target device, and for creating the release build.

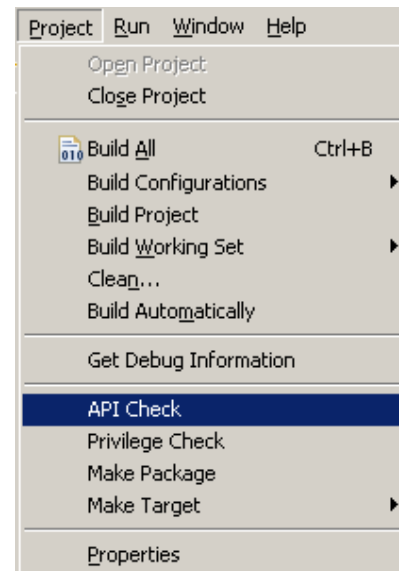


2. You can build your project using the **Build Project** menu item.



Checking API Violations

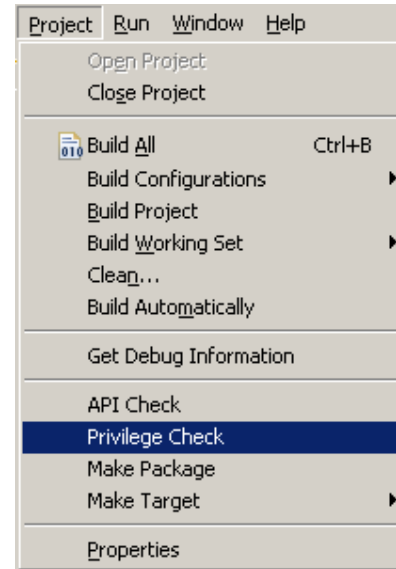
- You can check the source code in your project for any violation of APIs, such as incompatibilities between the defined API version and the used APIs:
 - Build the project.
 - Select **Project > API Check**.



- The check results are displayed in the **Problems** view.
- Note that the API check can also be performed automatically during the building process. For more information, see the Setting Application Properties slides in this tutorial.

Checking Privilege Violations

- You can check the source code in your project for any violation of privileges:
 1. Build the project.
 2. Select **Project > Privilege Check**.



- The check results are displayed in the **Problems** view.
- Note that the privilege check can also be performed automatically during the building process. For more information, see the Setting Application Properties slides in this tutorial.

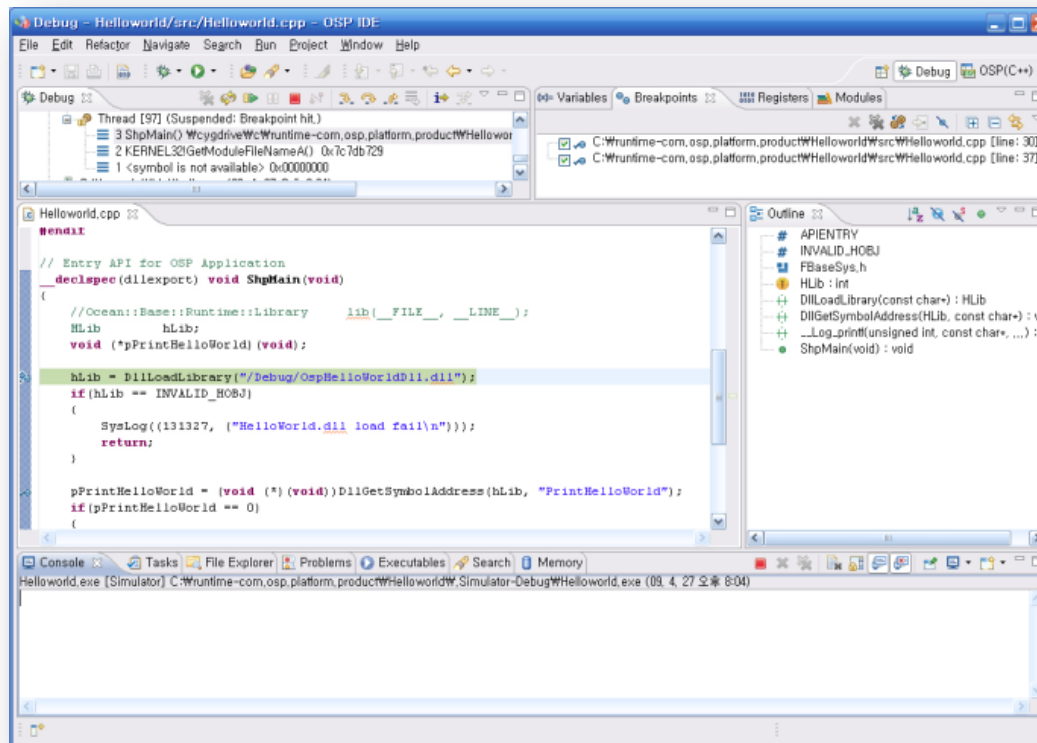
Debugging Applications (1/4)

- You can expect debugging to be essentially the same as with other development that you do. You can set breakpoints, step in, step out, and step over breakpoints, and watch variables.
- Debugging with bada IDE is same as with Eclipse CDT.
 - Refer to Help Contents: **C/C++ Development User Guide > Tasks > Running and debugging projects > Debugging.**
- The debugging environment uses GDB (GNU Debugger) for both the Simulator and target device. GDB can debug both locally and remotely.
- For more information about GDB, see <http://www.gnu.org/software/gdb/gdb.html>.



Debugging Applications (2/4)

- You can debug an application on the Simulator as well as the target device.



- To debug the application on the Simulator, right-click the project in the **Project Explorer** and select **Debug As > bada Simulator Application**.
- To debug the application on your target, right-click the project in the **Project Explorer** and select **Debug as > bada Target Application**.

Debugging Applications (3/4)

- The limitations for debugging in the target are:
 - Due to hardware limitations, you can insert only a limited number of breakpoints, as defined in the table below:

Model	Maximum number of breakpoints
Wave	4
WaveWQ	2

- You cannot insert watchpoints to monitor the variables used in your application.
- An application is suspended only after clicking the **Suspend** button followed by some other action in the application. For example, after clicking the **Suspend** button and some other button.



- The only latest application installed by bada IDE can be executed from the main menu.

Debugging Applications (4/4)

- Caution:
 - Do not step out during debugging inside of `OspMain()` in `<ProjectName>Entry.cpp`.
 - The program counter has problems when stepping out during debugging in `OspMain()` because `OspMain()` is the entry point for bada applications, and the upper call stack is the Windows `ntdll.dll` library.



Installing a Device Driver for Test Devices

- Extract `<model>-<driver>-<version>.zip` in `<BADA_SDK_HOME>\Tools` and run `Setup.exe`.
- Connect the test device to your computer with a USB cable.
- Select a folder for the driver location in the Hardware Update Wizard. The Hardware Update Wizard is shown twice for 2 modem ports.
 - Select **Install from a list or specific location (Advanced)**.
 - Check **Include this location in the search**.
 - Browse the folder with the device driver.
- Check if there are the 2 modem ports in Device Manager.
 - “Samsung Mobile Modem in V2” in **Modem**.
 - “Samsung Mobile Modem Diagnostic Serial Port V2 (WDM)” in **Port**.



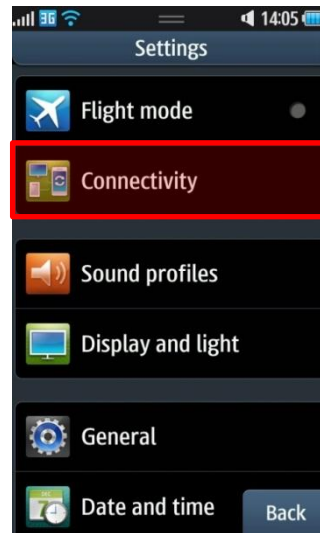
Installing the Test Root Certificate (1/4)

For testing an application on the target device during development, you must install the test root certificate to the target device. The application cannot be installed or run on the target device without the test root certificate.

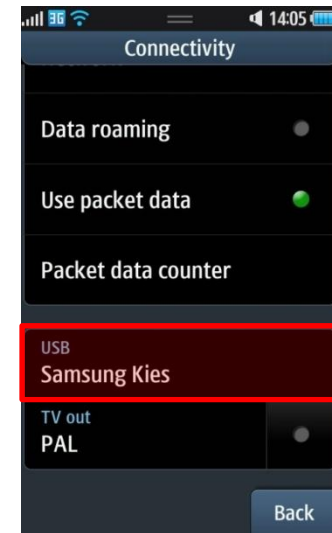
1. On the target device, tap **Settings** (A).
2. Tap **Connectivity** (B).
3. Tap **USB** (C).



A



B



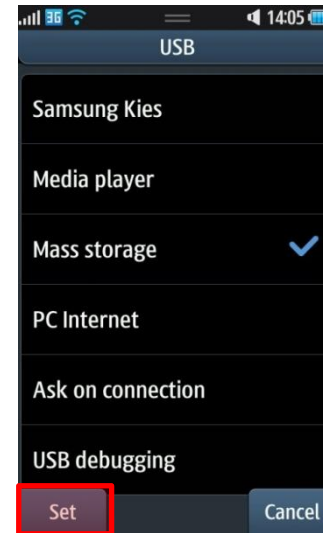
C

Installing the Test Root Certificate (2/4)

4. Tap **Mass storage** (D).
5. Tap **Set** (E).



D



E

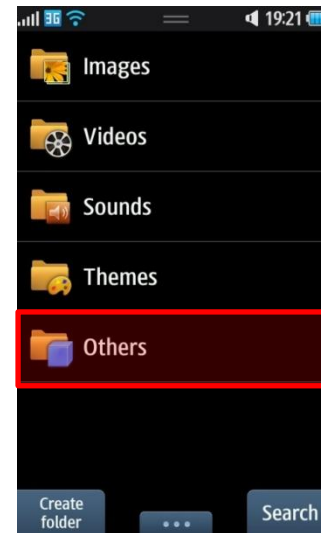
6. Connect the target device to your computer.

Installing the Test Root Certificate (3/4)

7. On your computer, copy the
`\<BADA_SDK_HOME>\Tools\sbuild\rootCACert.cer` file to
Removable Disk\Others.
 - The removable disk is connected to the target device.
8. Disconnect the target device from your computer.
9. On the target device, tap **My files** (F).
10. Tap **Others** (G).



F

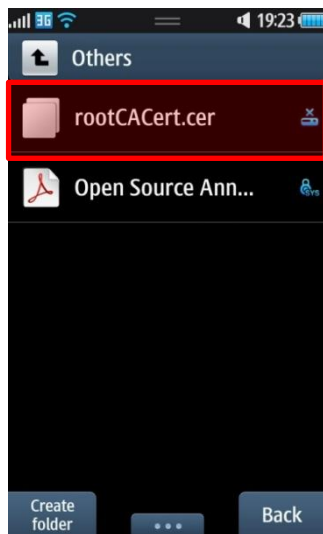


G

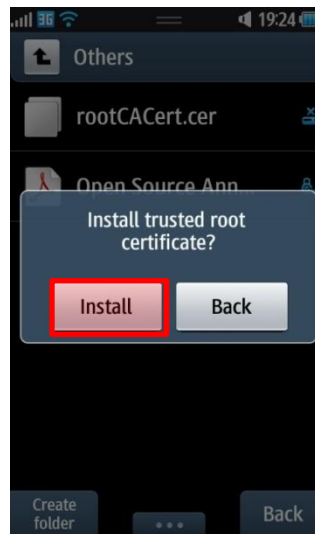
Installing the Test Root Certificate (4/4)

11. Tap **rootCACert.cer** (H).
12. Tap **Install** (I).

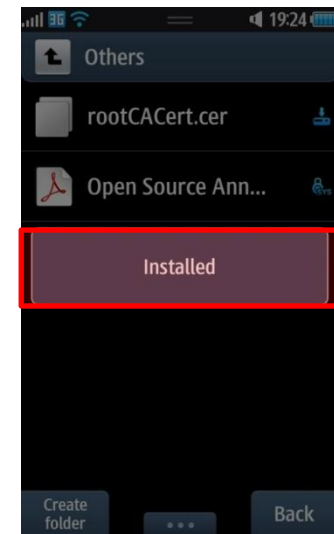
When the installation is completed, a message “Installed” is displayed on the screen (J).



H



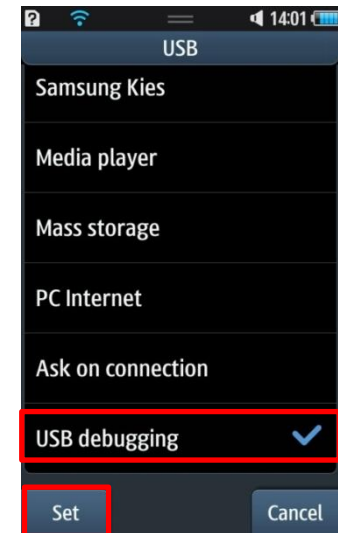
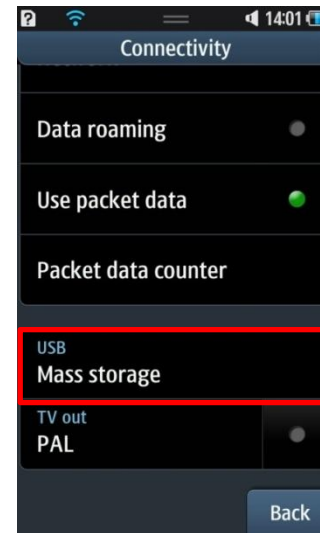
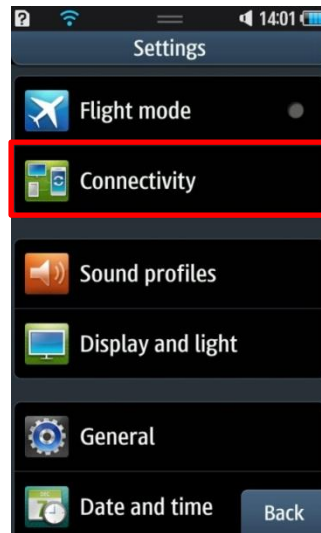
I



J

Setting the Target Device to Debug Mode

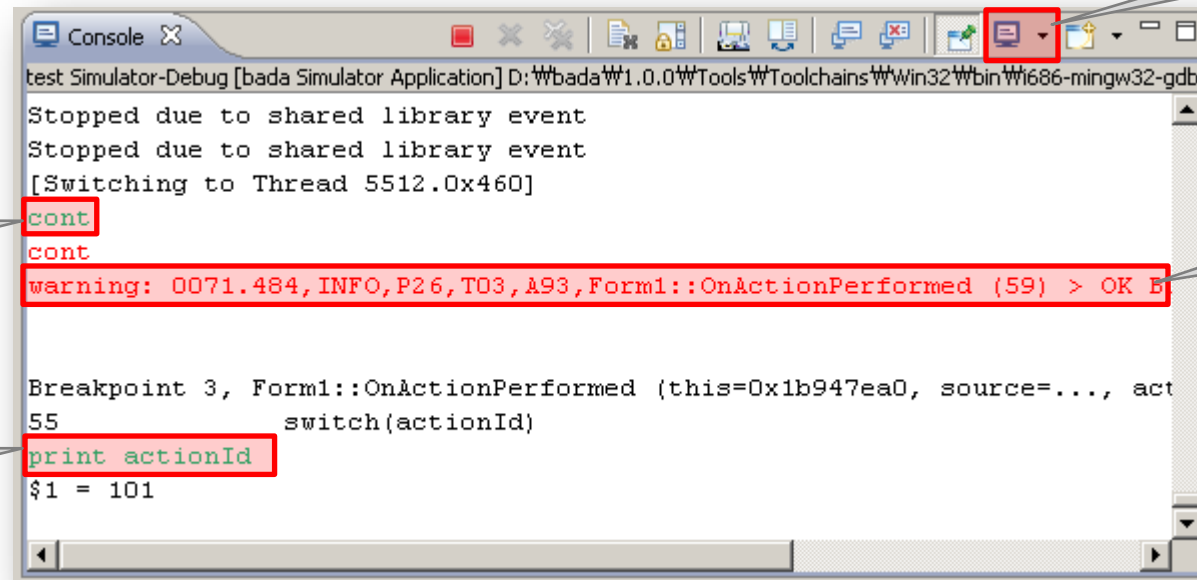
Set your target device to the debug mode by selecting **Settings > Connectivity > Mass storage > USB debugging**.



Checking Debug Messages (1/2)

You can check debug messages in the **Console** view.

- Click the **Display Selected Console** button and select the option containing **gdb**.
- With the GDB console, you can also execute GDB commands. For more information about GDB commands, see <http://www.gnu.org/software/gdb/gdb.html>.



The screenshot shows a GDB console window with the following content:

```
test Simulator-Debug [bada Simulator Application] D:\bada\1.0.0\Tools\Toolchains\Win32\bin\i686-mingw32-gdb
Stopped due to shared library event
Stopped due to shared library event
[Switching to Thread 5512.0x460]
cont
cont
warning: 0071.484, INFO, P26, T03, A93, Form1::OnActionPerformed (59) > OK B
Breakpoint 3, Form1::OnActionPerformed (this=0x1b947ea0, source=..., act
55          switch(actionId)
print actionId
$1 = 101
```

Display Selected Console button

GDB command

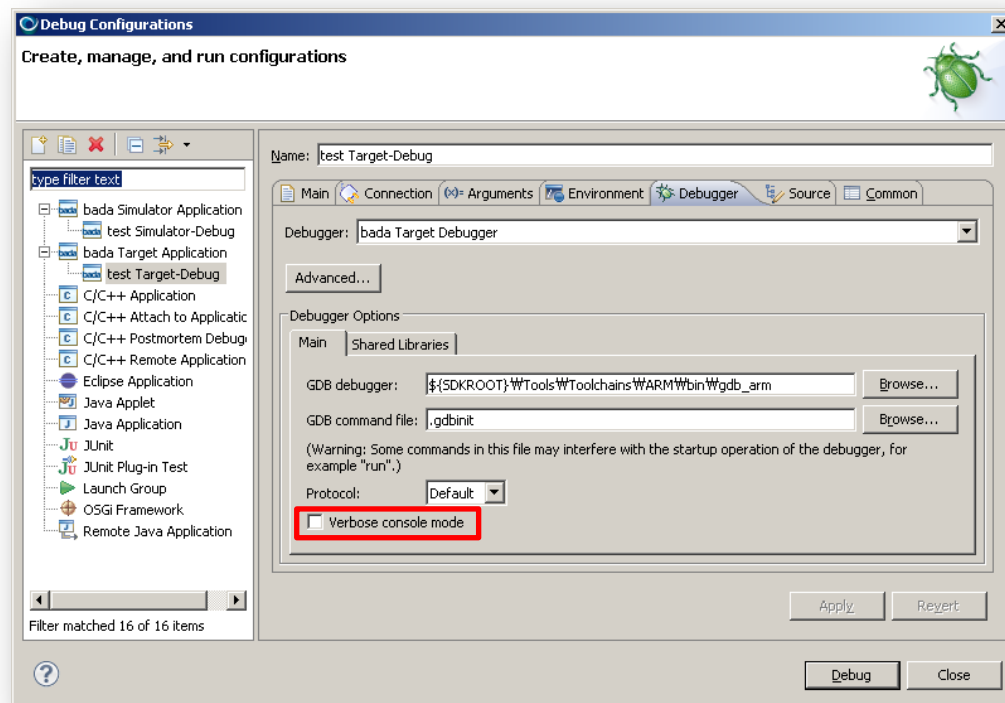
Log messages

GDB command

Checking Debug Messages (2/2)

You can also display verbose GDB output in the **Console** view:

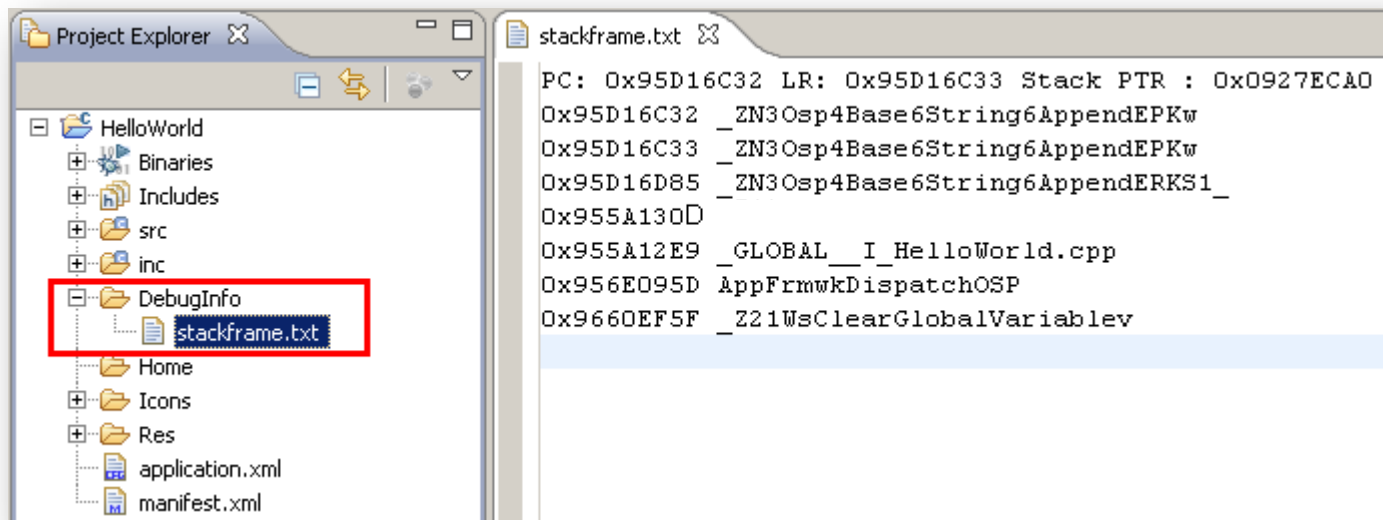
1. Right-click a project in the **Project Explorer** and select **Debug As > Debug Configurations....**
2. On the **Debugger** tab, select the **Verbose console mode** check box.
3. Click **Debug**.



Getting Debug Information

You can get debug information after the target terminates abnormally during debugging:

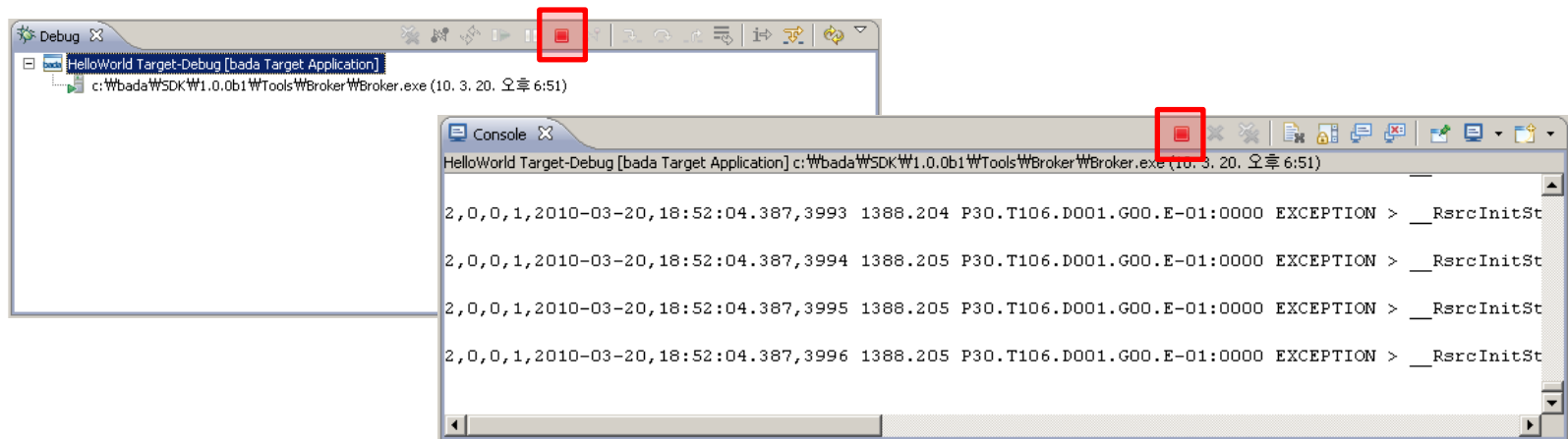
- Debug information includes stack frame.
- Select **Project > Get Debug Information**.



Stopping the Debug Process

To stop debugging:

- Click the **Terminate** button in the **Console** or **Debug** view of the bada IDE. This terminates the run operation.



- An application exits only when the termination of the run operation is followed by some other action in the application. For example, it exits when you click the **Terminate** button and then press some other button in the application.

Running Applications (1/3)

You can run an application on the Simulator or the target device:

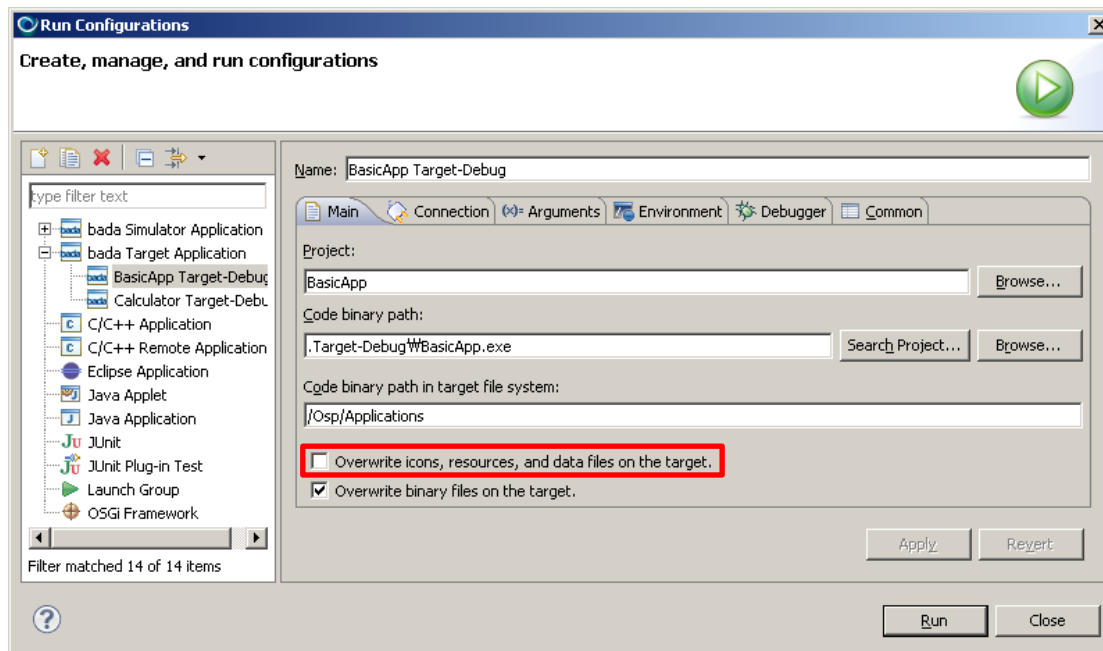
- The executable binary, resource files, and data files are downloaded before starting the run.
- The application is installed automatically while it runs.
- To run the application using the **Project Explorer**:
Right-click the project, and select **Run As > bada Simulator Application**, or **Run As > bada Target Application**.
- To run the application using the bada IDE menu:
Go to **Run > Run As > bada Simulator Application**, or **Run > Run As > bada Target Application**.



Running Applications (2/3)

If only the source code (not the resource files) has been changed, you can download the executable binary to your target without updating the other files.

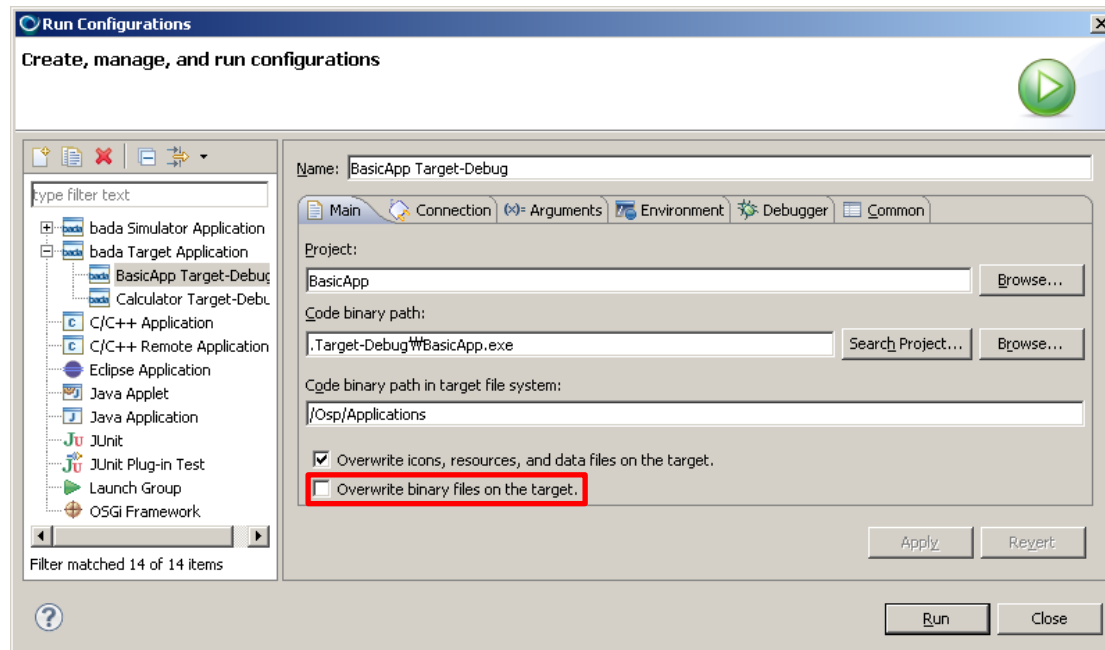
1. Right-click a project in the **Project Explorer** and select **Run As > Run Configurations....**
2. Clear **Overwrite icons, resources, and data files on the target.**
3. Click **Run.**



Running Applications (3/3)

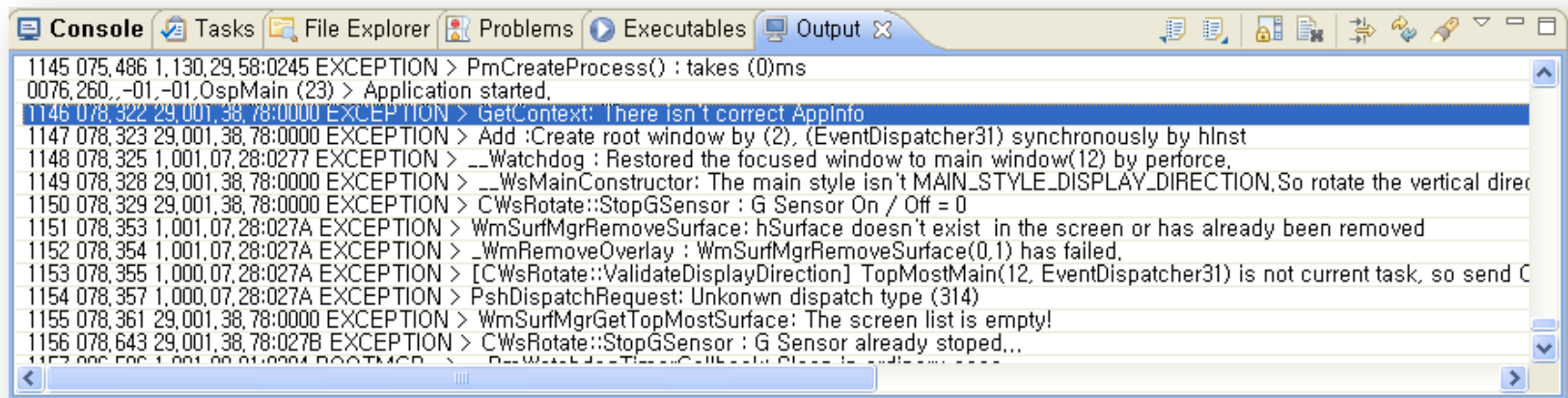
If only the resource files (not the source code) have been changed, you can download the modified files to your target without updating the binaries.

1. Right-click a project in the **Project Explorer** and select **Run As > Run Configurations....**
2. Clear **Overwrite binary files on the target.**
3. Click **Run.**



Checking Output Messages

- Start the **Output** view by selecting **Window > Show View > Other... > bada > Output**.
- The **Output** view functions similarly with the Simulator and the target device.

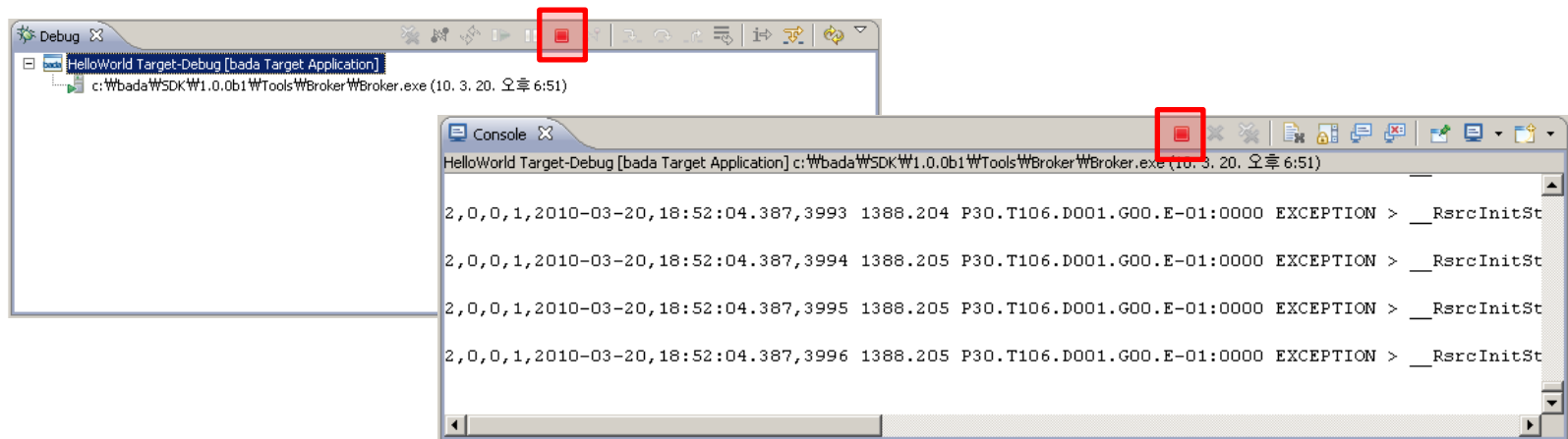


The screenshot shows the 'Output' window of an IDE. The window title bar includes 'Console', 'Tasks', 'File Explorer', 'Problems', 'Executables', and 'Output'. The main area contains a log of system messages, including several 'EXCEPTION' entries. The following table summarizes the visible log entries:

Time	Process	Message
1145 075,486 1,130,29,58:0245	PmCreateProcess()	: takes (0)ms
0076,260,-,-01,-01,OspMain (23)		> Application started.
1146 078,322 29,001,38,78:0000	EXCEPTION	> GetContext: There isn't correct AppInfo
1147 078,323 29,001,38,78:0000	EXCEPTION	> Add :Create root window by (2), (EventDispatcher31) synchronously by hInst
1148 078,325 1,001,07,28:0277	EXCEPTION	> _Watchdog : Restored the focused window to main window(12) by perforce.
1149 078,328 29,001,38,78:0000	EXCEPTION	> _WsMainConstructor: The main style isn't MAIN_STYLE_DISPLAY_DIRECTION, So rotate the vertical direc
1150 078,329 29,001,38,78:0000	EXCEPTION	> CWsRotate::StopGSensor : G Sensor On / Off = 0
1151 078,353 1,001,07,28:027A	EXCEPTION	> WmSurfMgrRemoveSurface: hSurface doesn't exist in the screen or has already been removed
1152 078,354 1,001,07,28:027A	EXCEPTION	> _WmRemoveOverlay : WmSurfMgrRemoveSurface(0,1) has failed,
1153 078,355 1,000,07,28:027A	EXCEPTION	> [CWsRotate::ValidateDisplayDirection] TopMostMain(12, EventDispatcher31) is not current task, so send C
1154 078,357 1,000,07,28:027A	EXCEPTION	> PshDispatchRequest: Unkonwn dispatch type (314)
1155 078,361 29,001,38,78:0000	EXCEPTION	> WmSurfMgrGetTopMostSurface: The screen list is empty!
1156 078,643 29,001,38,78:027B	EXCEPTION	> CWsRotate::StopGSensor : G Sensor already stopped...
1157 000,000 1,001,00,01:0000	EXCEPTION	> _Per-Watchdog-Timer-Callback: Class is un...

Exiting Applications

- To exit applications in the bada IDE:
 - Click the **Terminate** button in the **Console** or **Debug** view to terminate the run operation.

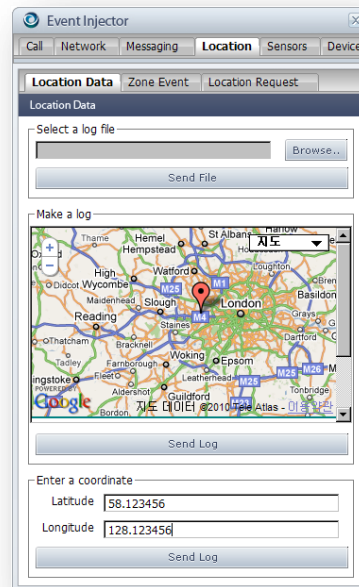


- An application exits only when the termination of the run operation is followed by some other action in the application. For example, it exits when you click the **Terminate** button and then press some other button in the application.
- To exit applications on the target device:
 - Press the **End** key on your target device to exit your application easily.

Testing Applications

- The Simulator provides a fast and efficient testing environment right on your local machine. You do not need to upload applications to target devices to run or test your applications.
- Debugging is done with native debuggers.
- The Simulator's Event Injector lets you artificially create and use any data you need. You can even simulate environmental conditions for device sensors:

- Call
- SMS
- Battery
- Antenna
- Sensors
 - Location
 - Accelerometer
 - Magnetic
 - Tilt
 - ...



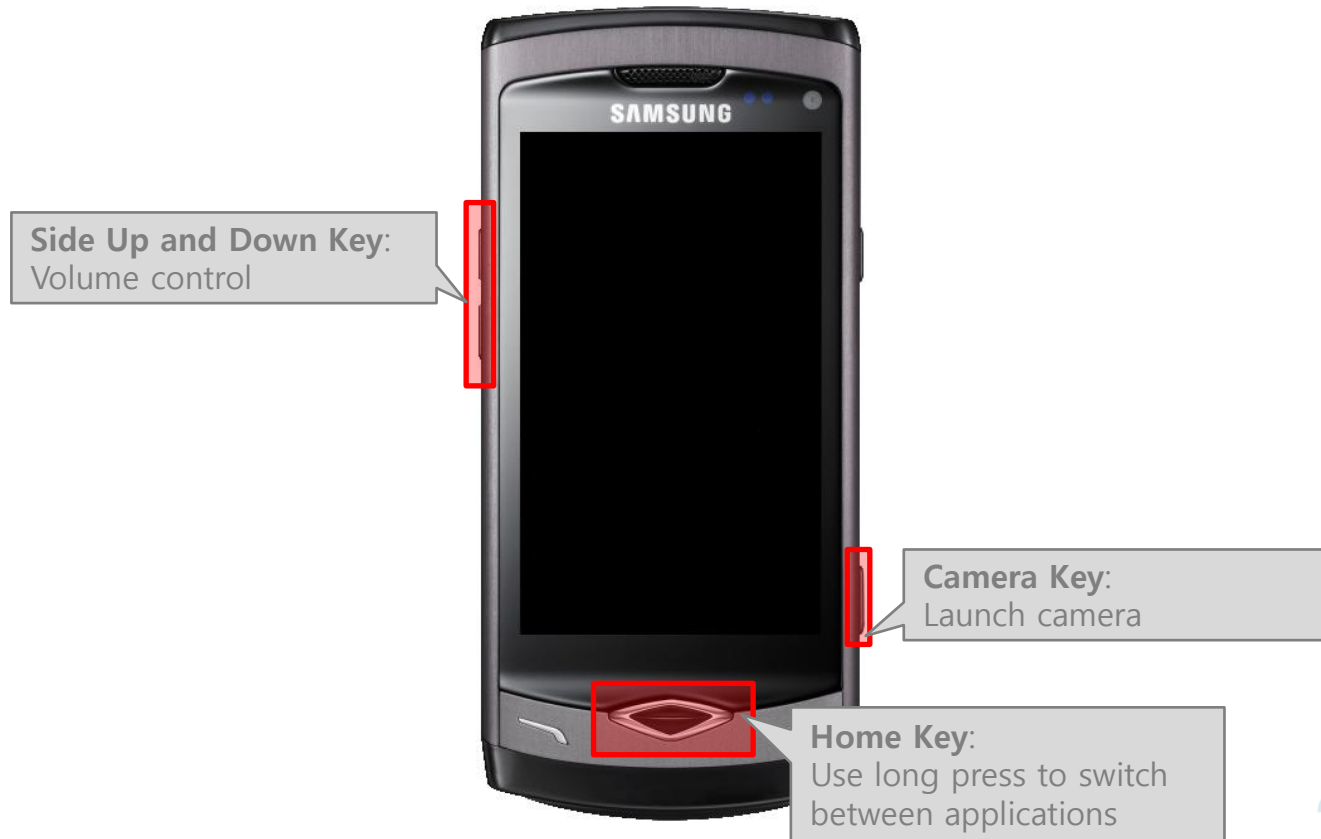
Inject arbitrary event data into device simulator



Note: The Simulator's Event Injector supports only English.

Manipulating Simulator (1/2)

- The Simulator is used in developing applications as it gives you execution control using native debuggers.



Manipulating Simulator (2/2)

The Simulator menu lets you easily control the Simulator environment.

- **Event Injector:** Launch the Event Injector.
 - For detailed information about the Event Injector, see the bada Developer Guide.
- **Rotate:** Switch between portrait and landscape mode.
 - The application must implement this feature in order to support rotation.

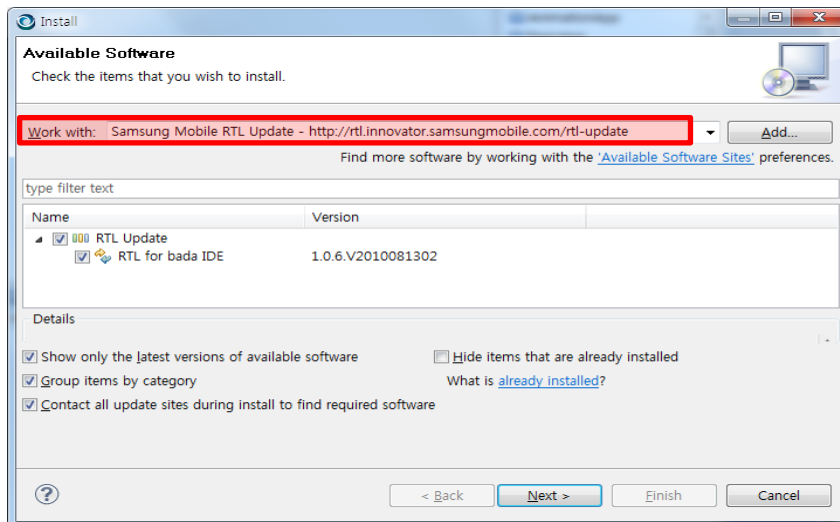


- **Error Lookup:** Retrieve a system error message based on the value entered.
- **Update Content DB:** Register the newly added files in the Media folder. (This option only works on the idle screen.)
- **Options:**
 - **Always on Top:** Set the z-order to the top.
 - **Size:** Adjust the Simulator size.



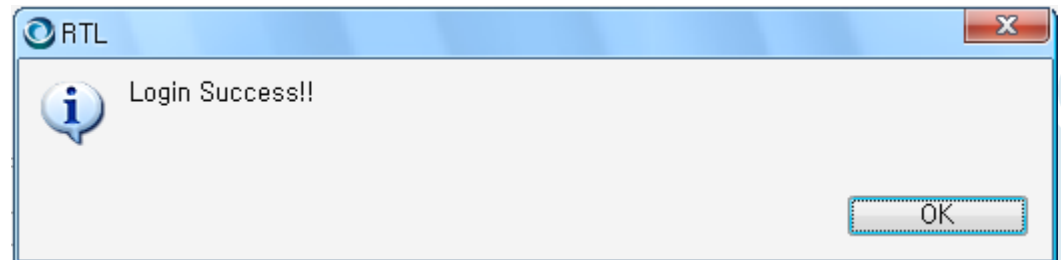
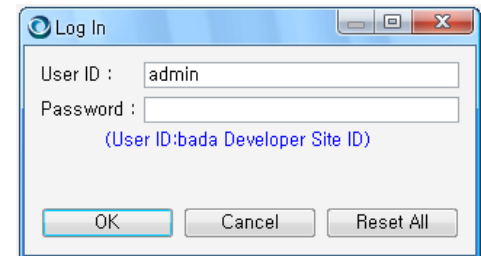
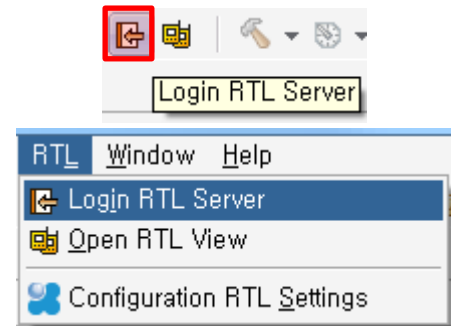
Testing Applications with Remote Test Lab (1/6)

- Remote Test Lab (RTL) allows you to install and test applications on Samsung devices over the Web. With RTL, you can run applications with a real device instead of the Simulator or local device.
- To use RTL, you must install the RTL Eclipse plug-in.
 1. In the bada IDE, go to **Help > Install New Software....**
 2. Select <http://rtl.innovator.samsungmobile.com/rtl-update> as the software site you work with.
 3. Select the plug-ins to install and click **Next** to complete the installation process.



Testing Applications with Remote Test Lab (2/6)

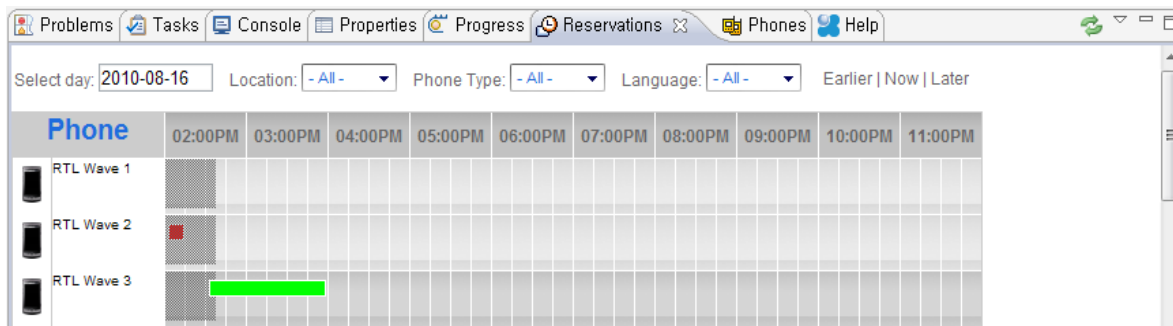
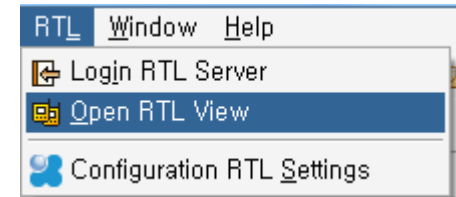
- To start testing applications, you must log in to RTL:
 1. Do one of the following:
 - Click the **Login RTL Server** button.
 - Go to **RTL > Login RTL Server**.
 2. Enter your user ID and password of the bada developer site (<http://developer.bada.com>).
RTL is available to developer site members free of charge (regardless of the membership level).
 3. After your login succeeds, a notification message is displayed.



Testing Applications with Remote Test Lab (3/6)

- To test applications, you must make a reservation for at least one device in RTL:

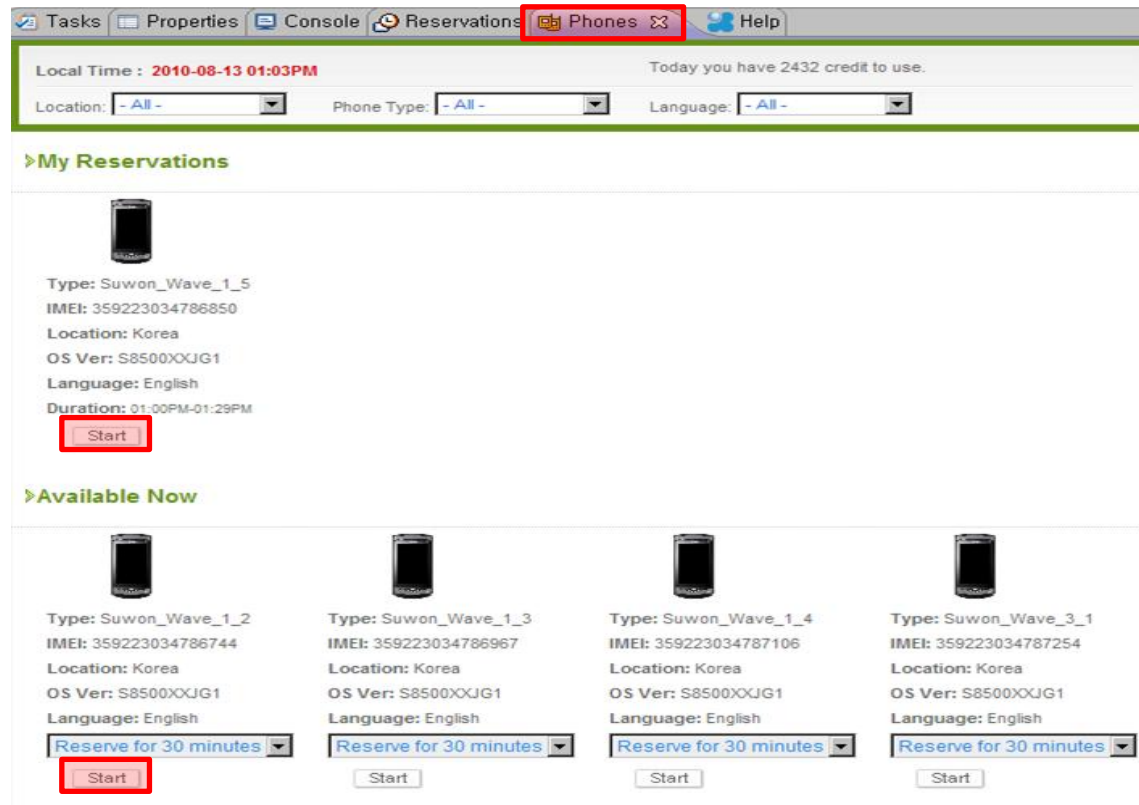
- Go to **RTL > Open RTL View**.
The **Reservations**, **Phones** and **Help** views open.
- In the **Reservations** view, select a device and drag and drop the bar in the time table to select the time period you want to reserve.



- To ensure fair access to all members, RTL uses a credit system for the reservations:
 - Every member receives 10 credits/day.
 - One credit buys 15 minutes in RTL.
 - The minimum period is 30 minutes (2 credits).
 - The maximum period is 10 hours/day (40 credits).
- For more information, go to **RTL > Open RTL View > Help View**.

Testing Applications with Remote Test Lab (4/6)

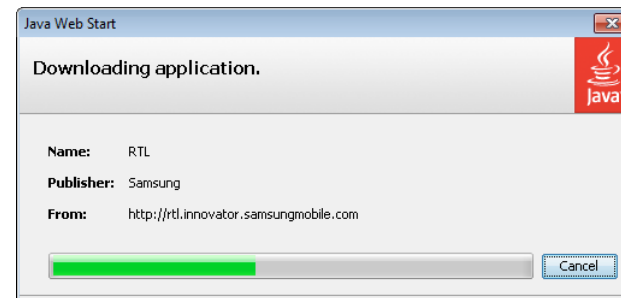
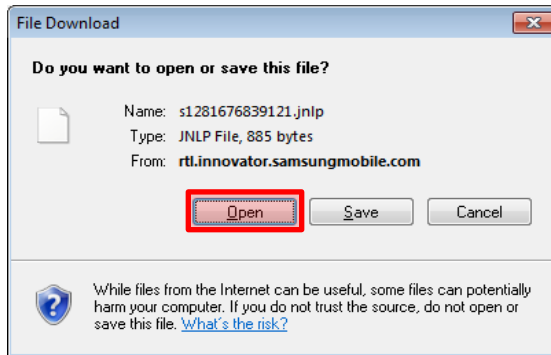
- To start the RTL client:
 - Go to **RTL > Open RTL View > Phones View**.
 - Select a device under **My Reservations** or **Available Now** and click **Start**.



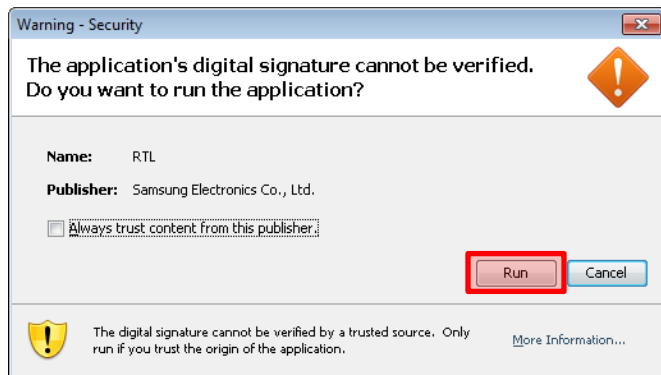
- After the client starts, you can install and run packaged applications (.zip files) on the device through the client.

Testing Applications with Remote Test Lab (5/6)

- Starting the RTL client invokes Java Web Start.
 - Click **Open** to start downloading the client module (runnable jar).



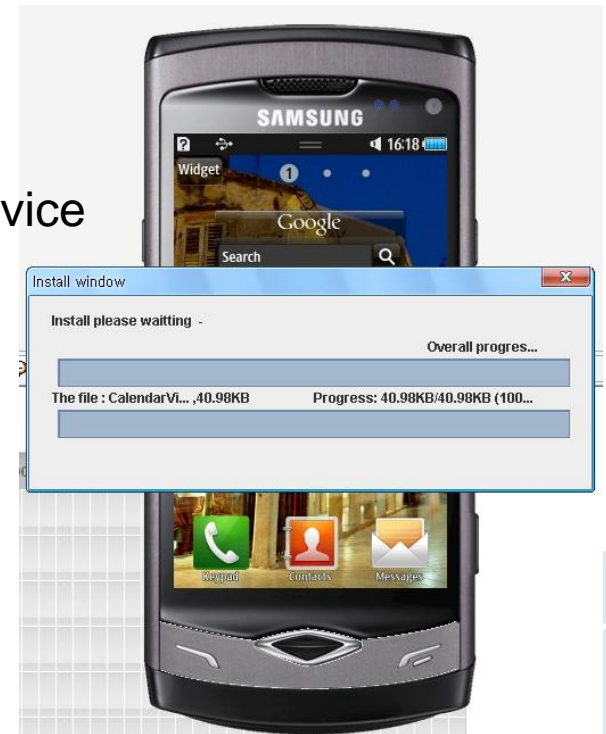
- Once the download completes, click **Run** to start the application.



- For more information about Java Web Start, see <http://java.sun.com>.

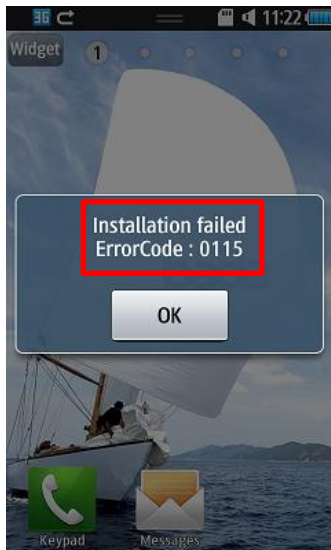
Testing Applications with Remote Test Lab (6/6)

- After reserving devices, you can also run your application projects from the IDE directly without first packaging them and starting the RTL client:
 1. Build your project with the **Target-Release** build configuration.
 2. Right-click a project in the **Project Explorer** and select **Run as > Run Configurations**.
 3. On the **Run Configurations** window, go to the **Connection** tab and select a reserved device from the drop-down list.
 4. Click **Run**.
The RTL client starts automatically.
- The executable binary, resource files, and data files are downloaded before starting the run.
- The application is installed automatically while it runs.



Error Codes (1/4)

- An error code is assigned to every error.
- Use the code to detect the nature of the error.



Error Codes (2/4)

The following table gives a detailed description for each error code:

Error code	Description
0101	Installation aborted: Another installation is in progress. Multiple applications cannot be installed at the same time.
0102	Installation failed: Invalid package. The application cannot be installed properly because some of the package contents are invalid or missing.
0103	Installation failed: Maximum number of applications reached. The number of applications that are installed has reached the maximum limit. Uninstall some applications first and try again.
0104	Installation failed: Insufficient storage. Free some memory and try again. There is insufficient storage space available for installing this application. Free some memory by removing unwanted media files or by uninstalling existing applications.
0105	Installation failed: Certificate Parsing Error. The certificate is invalid in Signature.xml. To perform the application integrity check, the derived certificate must be checked against the parent certificate before the actual installation.
0106	Installation failed: Certificate has expired. The certificate used to sign the application is expired.
0107	Installation failed: Certificate is revoked. The certificate used to sign the application is revoked according to the result of the OCSP protocol..
0108	Installation failed: Failed to find root certificate. The root certificate is not found on the device. The root certificate is required to build the certificate chain for the application integrity check.

Error Codes (3/4)

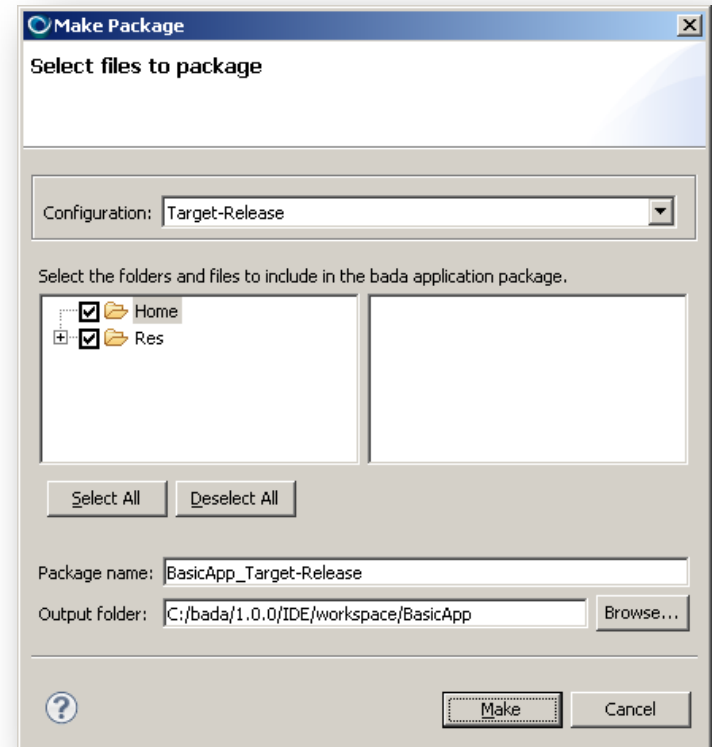
Error code	Description
0109	Installation failed: Signature verification failed. Failed to verify the application signature.
0110	Installation failed: Application metadata registration failed. Application metadata required to verify the application integrity is invalid or missing.
0111	Installation failed: Application signature missing or invalid. Application signature required to verify the application integrity is invalid or missing.
0112	Installation failed: Application manifest missing or invalid. The application manifest file specifying the application metadata is invalid or missing.
0113	Installation failed: Invalid application executable file. The executable binary of the application is invalid or missing.
0114	Installation failed: Invalid application information. The application information file is invalid or missing. This file specifies application details such as the application name and path to application icons.
0115	Installation failed: Privilege registration failed. Failed to register the privilege information to the system database during installation.
0116	Installation failed: Unknown Error The application installation failed due to an unknown reason.
0117	Installation failed: Main menu icon missing. The application's main menu icon is missing.
0118	Installation failed: Setting menu icon missing. The application's Setting menu icon is missing.
0119	Installation failed: Ticker icon missing. The application's ticker icon is missing.
0120	Installation failed: Quick panel icon missing. The application's quick panel icon is missing.

Error Codes (4/4)

Error code	Description
0121	Installation failed: Launch image missing. The application's launch image is missing.
0122	Installation failed: Main menu icon name invalid. The file name of the main menu icon cannot be extracted properly from the application package.
0123	Installation failed: Setting menu icon name invalid. The file name of the Setting menu icon cannot be extracted properly from the application package.
0124	Installation failed: Ticker icon name invalid. The file name of the ticker icon cannot be extracted properly from the application package.
0125	Installation failed: Quick panel icon name invalid. The file name of the quick panel icon cannot be extracted properly from the application package.
0126	Installation failed: Launch image name invalid. The file name of the launch image cannot be extracted properly from the application package.
0127	Installation failed: License acquisition failed. Failed to acquire the license during the DRM file installation.
0128	Installation failed: Failed to decrypt DRM package. Failed to decrypt the DRM package with the license file for some reason, such as 'key not found'.
0201	Launch error: No License. The license required to launch the application is missing or the PNUM does not match.
0202	Launch error: Signature verification failed. Failed to verify the signature because the application binary or hash table file has been modified after installation.
0203	Launch error: File(s) missing for signature verification. The files required to check the hash table are missing.

Packaging Applications

- You can create a new compressed bada application package to register in Samsung Apps.
- Go to **Project > Make Package**.
 1. Select **Target-Debug** or **Target-Release** as a configuration.
 2. Select the files and folders to be included in the package.
 3. Enter a name for the package and select an output folder.
- Click the **Make** button.
 - Check the created package in the output folder.
 - **Note:** The IDE cannot directly install the created package to a target device.



Packaging Applications on Command Line

You can also create a new compressed bada application package without the IDE by using `MakePackage.exe` through the command line.

- File location:

`<BADA_SDK_HOME>\Tools\MakePackage\MakePackage.exe`

- Command syntax:

`MakePackage.exe TYPE_OPTION PROJECT_PATH`

where `TYPE_OPTION` can be:

- `-s` : Simulator-Debug
- `-td` : Target-Debug
- `-tr` : Target-Release

- Command example:

- `MakePackage.exe -tr C:\MyWorkspace\MyProject`



Upgrading Applications

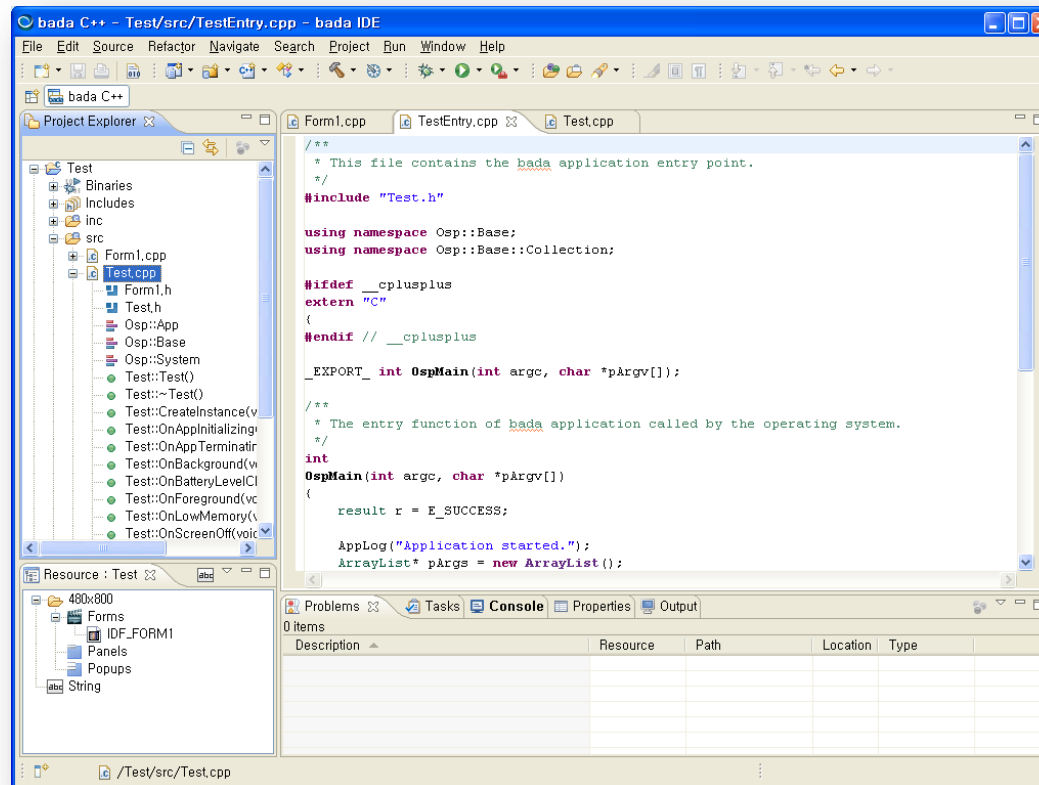
- If you need to upgrade your application after it has been certified at Samsung Apps for sales, you must:
 1. Download an updated manifest file from the [bada developer site](#).
 2. Repackage the application.
 3. Register the upgraded application on Samsung Apps Seller Office.
- When a previously installed application is upgraded on the device, you can decide which data files from the old version are kept and which are deleted.

The common bada upgrade policy is to overwrite all the application package files, while keeping the user-created files and directories.

Application project folder	Effect of the upgrade
/Home	Application package files are overwritten. User-created files and directories are not deleted.
/Res	Packaged resource files are read-only. They are replaced with the newly packaged resource files.
/Bin	Application binaries are read-only. They are replaced with the newly packaged application binaries.

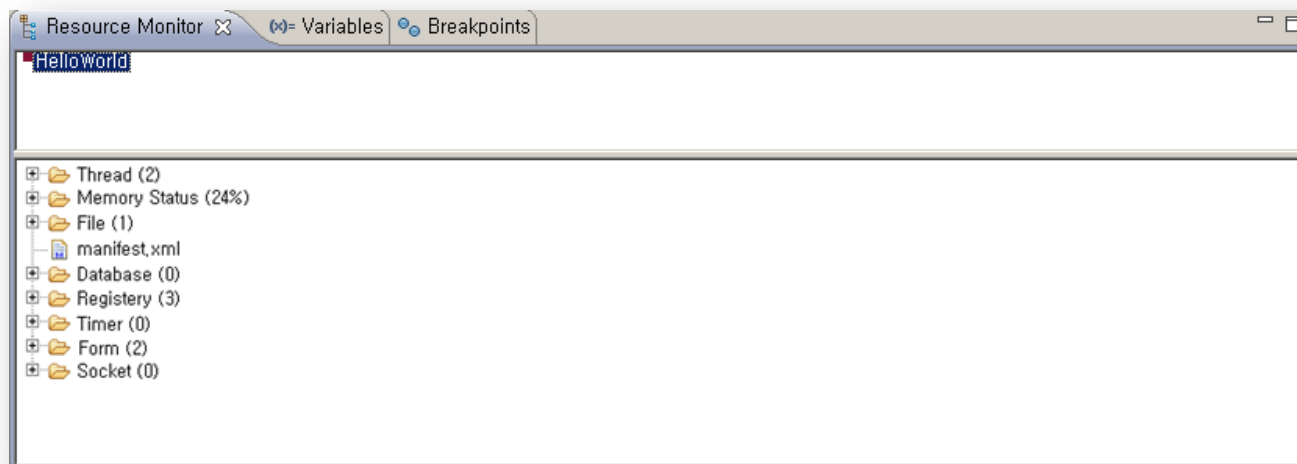
IDE Text Editor

Eclipse provides a wealth of code editing tools in its text editor. All the CDT features are available, including syntax highlighting, code folding, tabbed documents, and content assist (context-sensitive code completion).



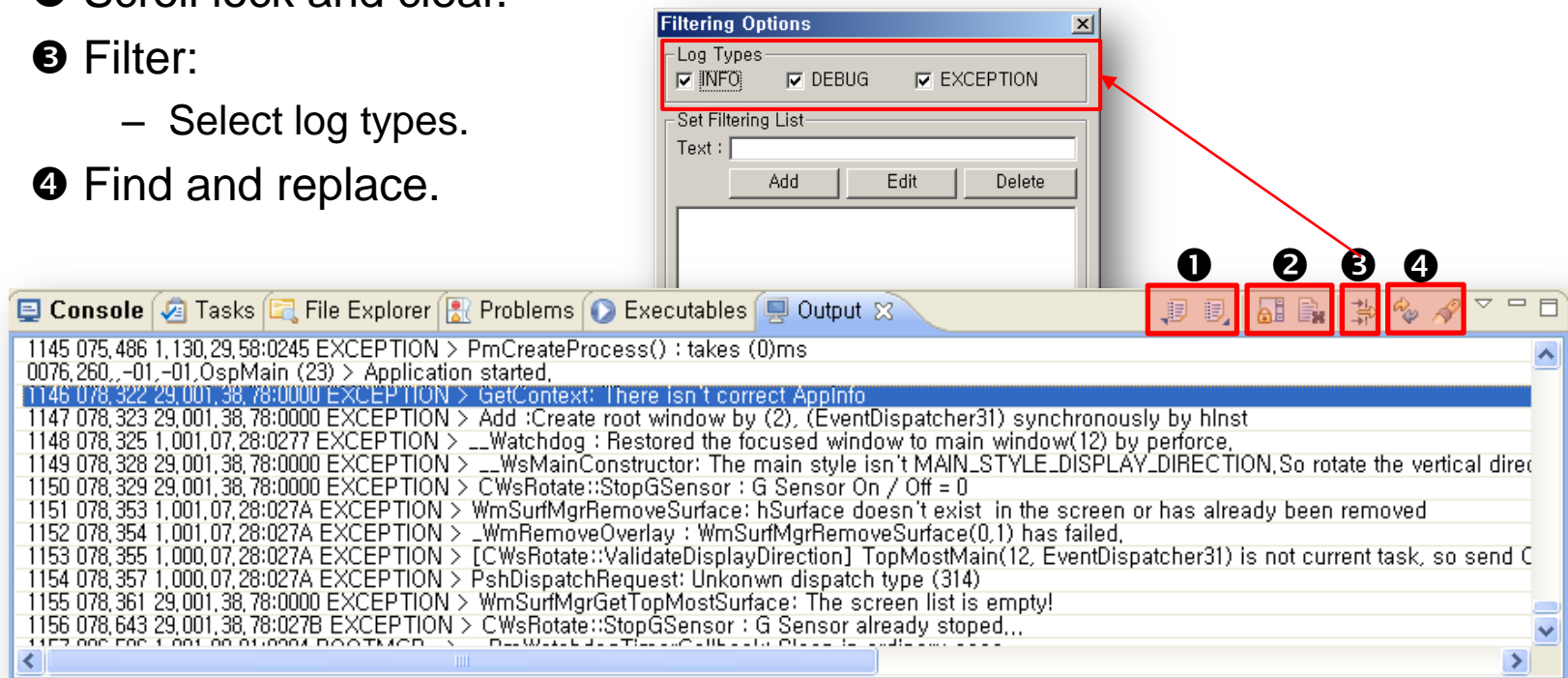
Resource Monitor

- Start the Resource Monitor by selecting **bada – Resource Monitor** under **Window > Show View > Other...**
- You can monitor resource usage statistics to help you create efficient applications. Available monitoring features include:
 - Running applications in the Simulator or a target device.
 - Thread, heap, file, database, registry, timer, form, and socket information.
 - Memory leak information.
- You cannot monitor resource usage during target debugging.

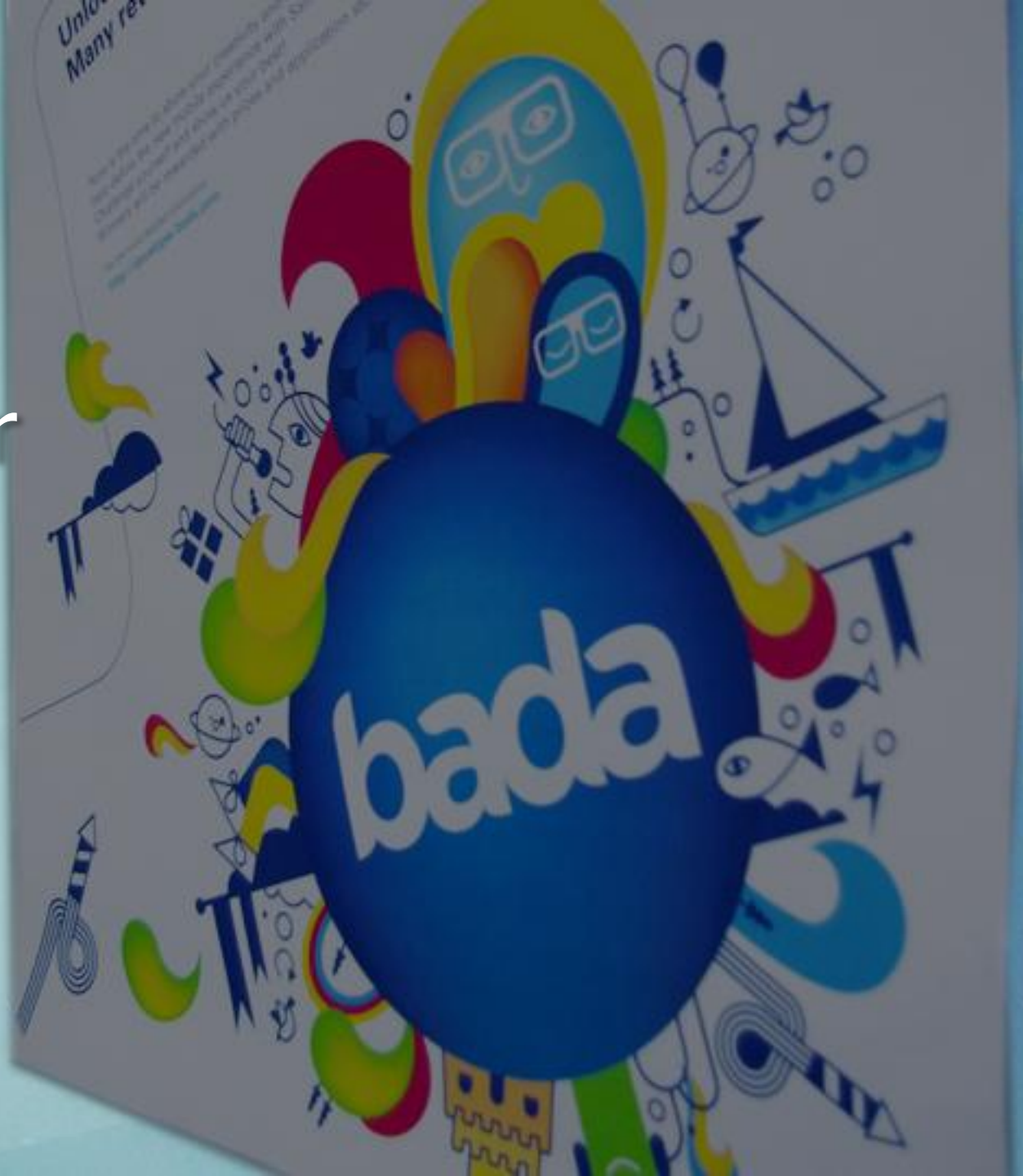


Output

- Start the **Output** view by selecting **Window > Show View > Other... > bada > Output**.
- The **Output** view shows the log, debug, and exception messages from `AppLog()`, `AppLogDebug()` and `AppLogException()`.
 - ❶ Import and export to and from log files.
 - ❷ Scroll lock and clear.
 - ❸ Filter:
 - Select log types.
 - ❹ Find and replace.



UI Builder



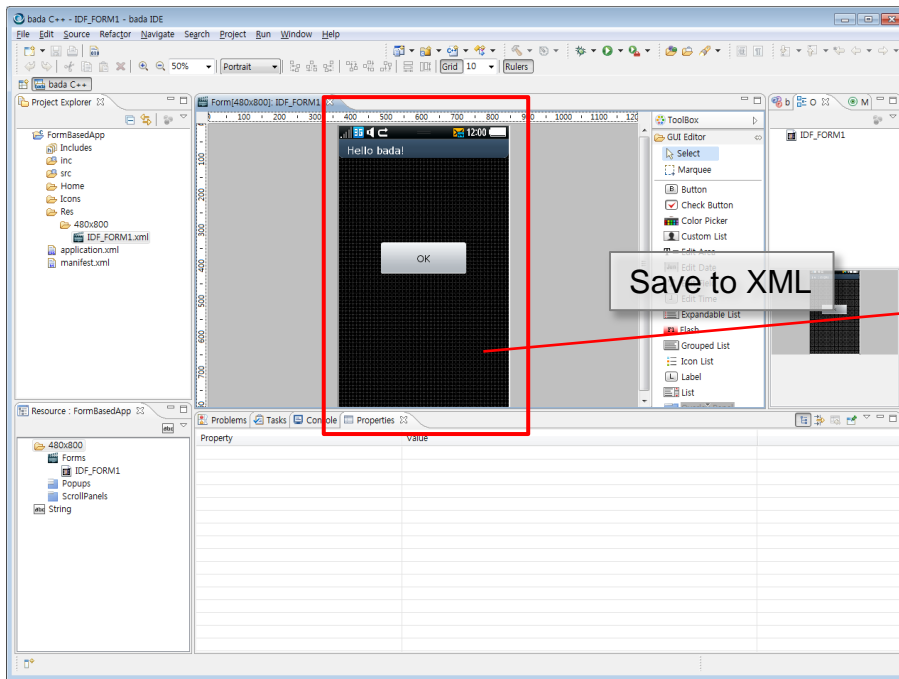
Contents

- WYSIWYG Design with UI Builder
- UI Builder Toolbar
- Using Forms
 - Form Properties
 - Dragging and Dropping Controls
 - Designing Landscape Forms
 - Creating Form Classes for the Designed Forms
 - Constructing Forms
- Using Scroll Panels
- Using Pop-ups
- Using String Tables
- Setting Display Language
- Adding New Keyboard Languages



WYSIWYG Design with UI Builder

You can create compelling GUIs with WYSIWYG design tools.



```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<!DOCTYPE Scene SYSTEM "UIForm.dtd">
<Scene>
  <Form id="IDF_FORM1">
    <property title="" headerHeight="0" ... />
    <layout mode="Portrait" style="FORM_STYLE_INDICATOR|..." />
    <layout mode="Landscape" style="FORM_STYLE_NORMAL" />
  </Form>
  <Button id="IDC_BUTTON1" parent="IDF_FORM1">
    <property text=""
      hAlign="ALIGN_CENTER" vAlign="ALIGN_MIDDLE"
      normalBGColor="" normalTextColor=""
      pressedBGColor="" pressedTextColor=""
      disableBGColor="" disableTextColor="" />
    <layout mode="Portrait"
      x="40" y="526" width="410" height="90" />
    <layout mode="Landscape"
      x="66" y="352" width="683" height="54" />
  </Button>
</Scene>
```

UI Builder Toolbar

In addition to regular keyboard and mouse control, the form designer toolbar provides additional time-saving design tools.

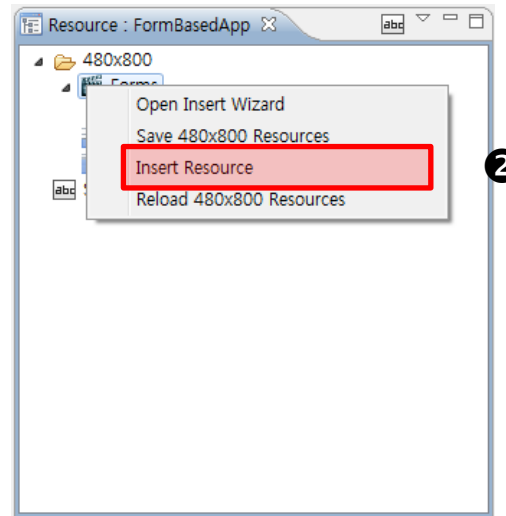
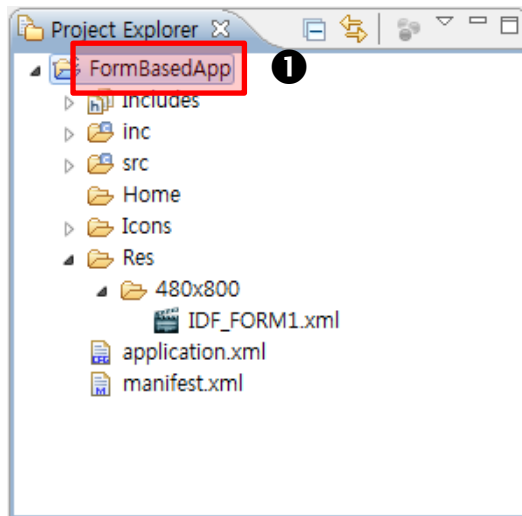
– Toolbar:

- ① Zoom
- ② Editing mode: Portrait, Landscape
- ③ Align controls
- ④ Toggle grid
- ⑤ Toggle rulers



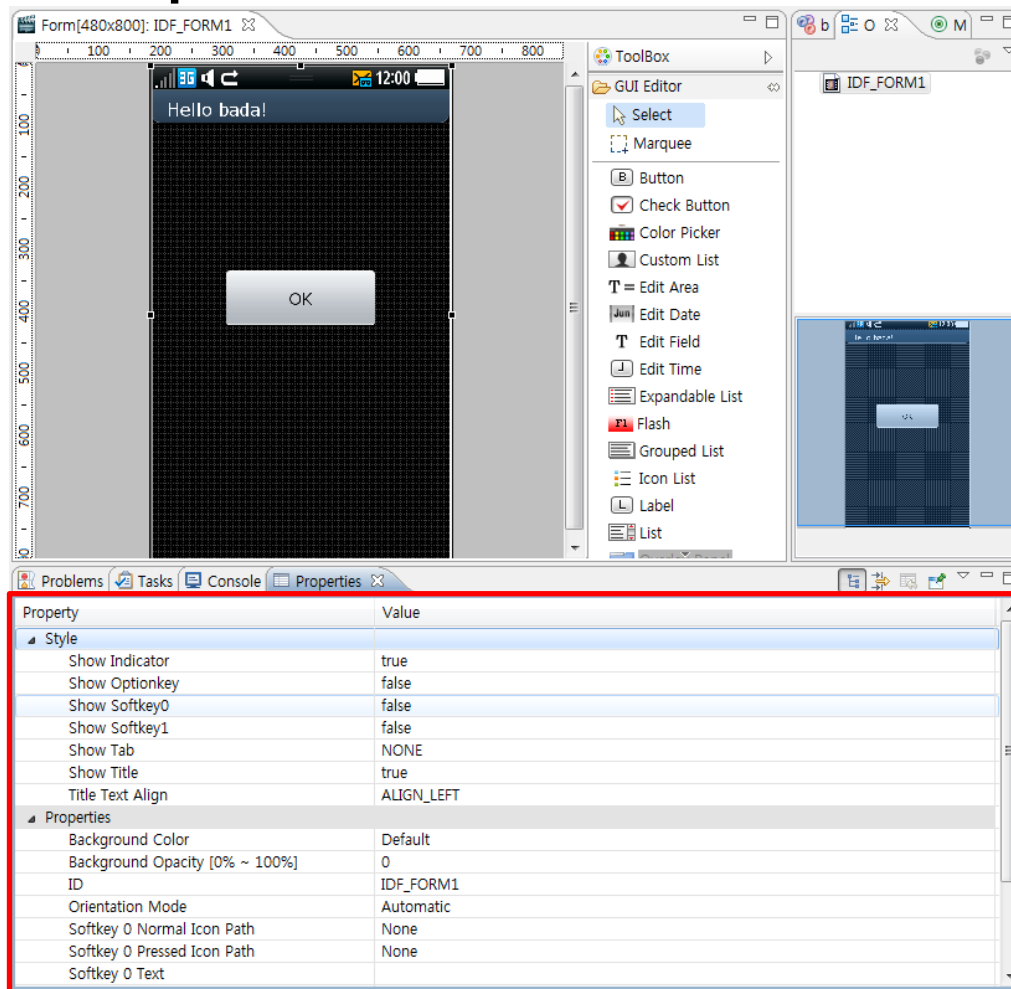
Using Forms

- Go to **Window > Show View > Other...** and select **bada Resource Explorer** if you do not see the **Resource Explorer**.
- Adding forms to your application is easy:
 - ❶ Select a project in the **Project Explorer**.
 - ❷ Right-click to add a new form:
 - Select **Insert Resource** in the context menu.
 - Double-click the new form to open it in the editor.



Form Properties

You can add an **Indicator bar**, **Title bar**, **Soft key**, and **Option key** to the form in the **Properties** tab.



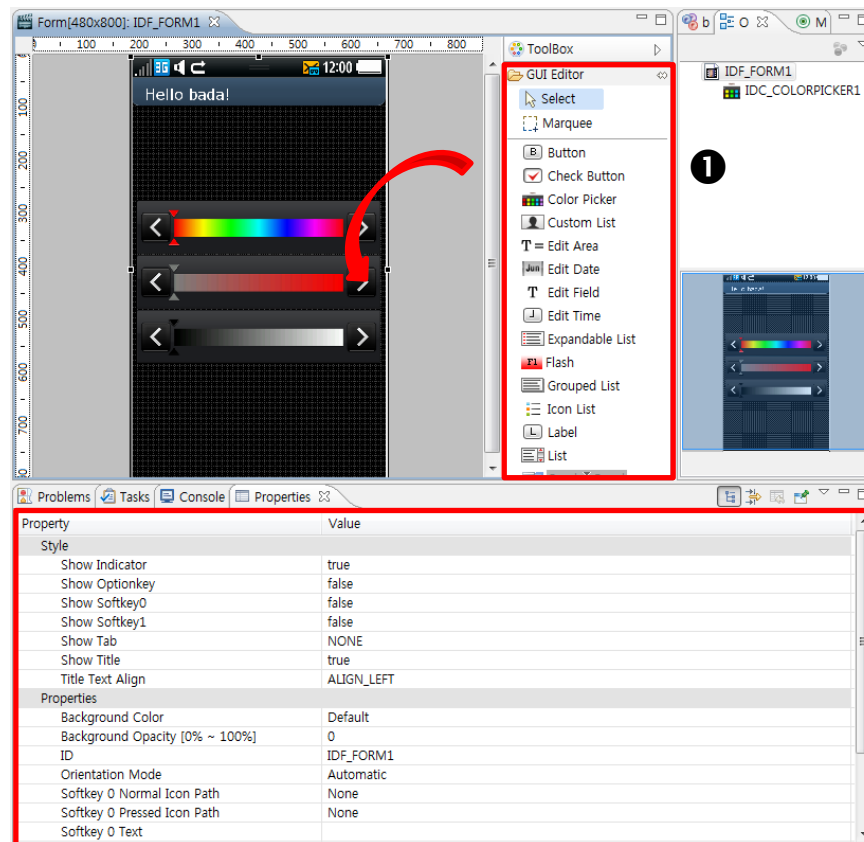
The screenshot shows the GUI Editor interface for a form named 'IDF_FORM1'. The form is displayed in a dark theme with a grid background. It features a status bar at the top with the text 'Hello bada!' and a time display of '12:00'. Below the status bar is a large text area containing the text 'OK'. The Properties tab is open, showing the following properties:

Property	Value
Style	
Show Indicator	true
Show Optionkey	false
Show Softkey0	false
Show Softkey1	false
Show Tab	NONE
Show Title	true
Title Text Align	ALIGN_LEFT
Properties	
Background Color	Default
Background Opacity [0% ~ 100%]	0
ID	IDF_FORM1
Orientation Mode	Automatic
Softkey 0 Normal Icon Path	None
Softkey 0 Pressed Icon Path	None
Softkey 0 Text	

Dragging and Dropping Controls

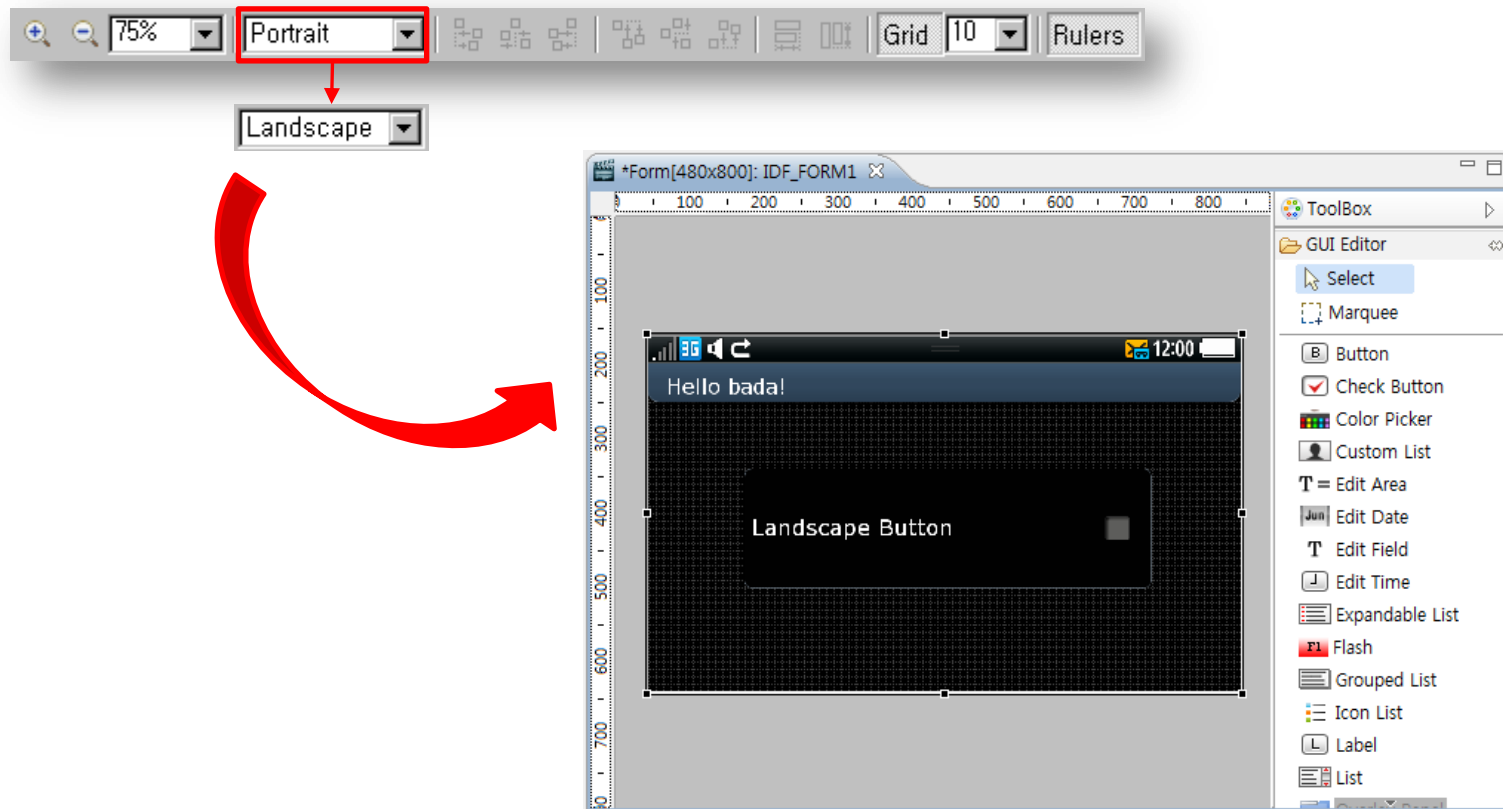
Add in additional GUI controls by dragging and dropping them onto your application form and setting control properties:

- 1 Select a control in the controls palette and drag and drop it onto the form.
- 2 Set the control's properties.



Designing Landscape Forms

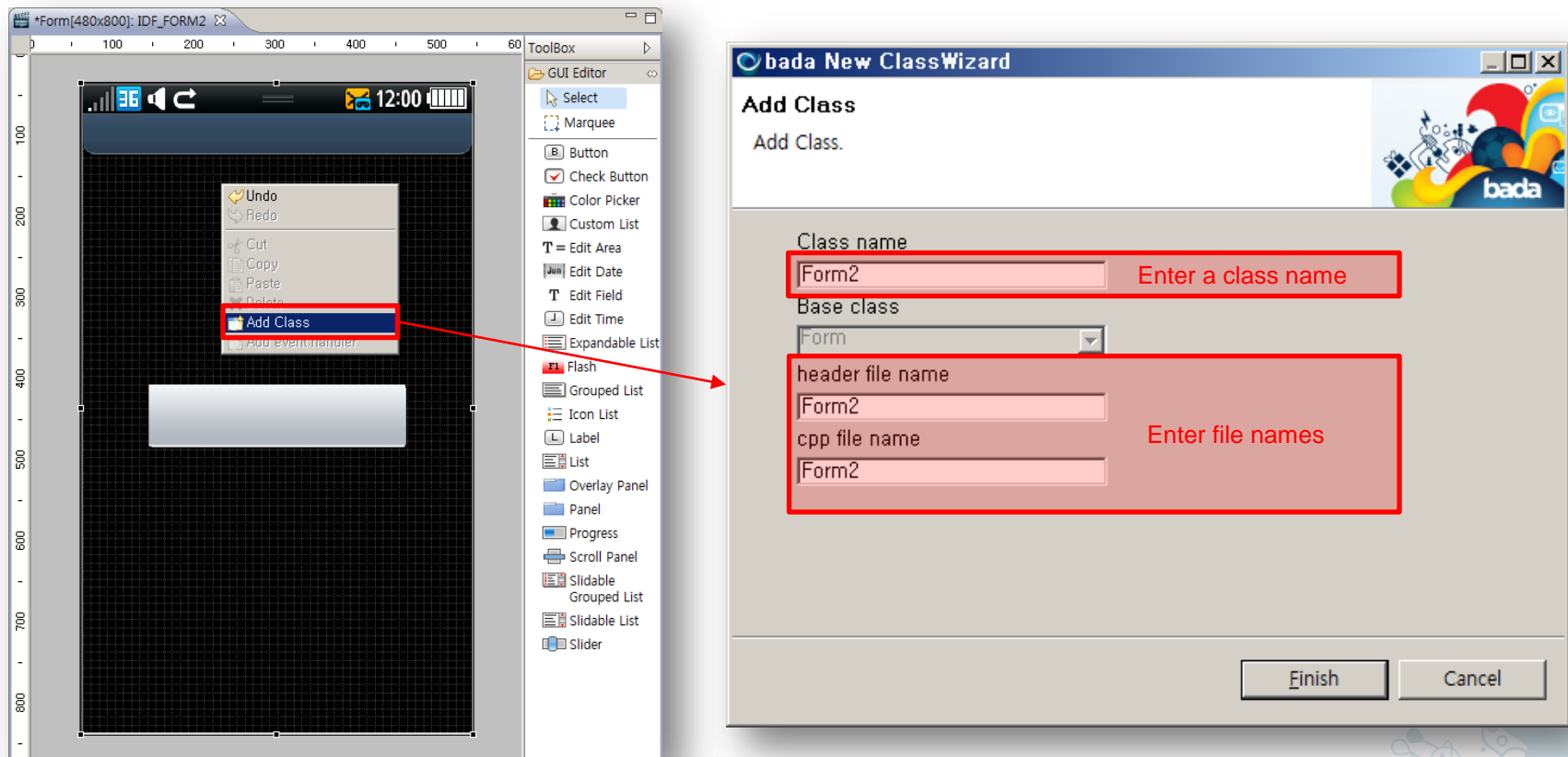
You can also create forms in landscape mode by selecting **Landscape** in the designer toolbar. This keeps the proper screen-size for your target device.



Creating Form Classes for the Designed Forms (1/2)

You can add new forms to your application by creating new classes:

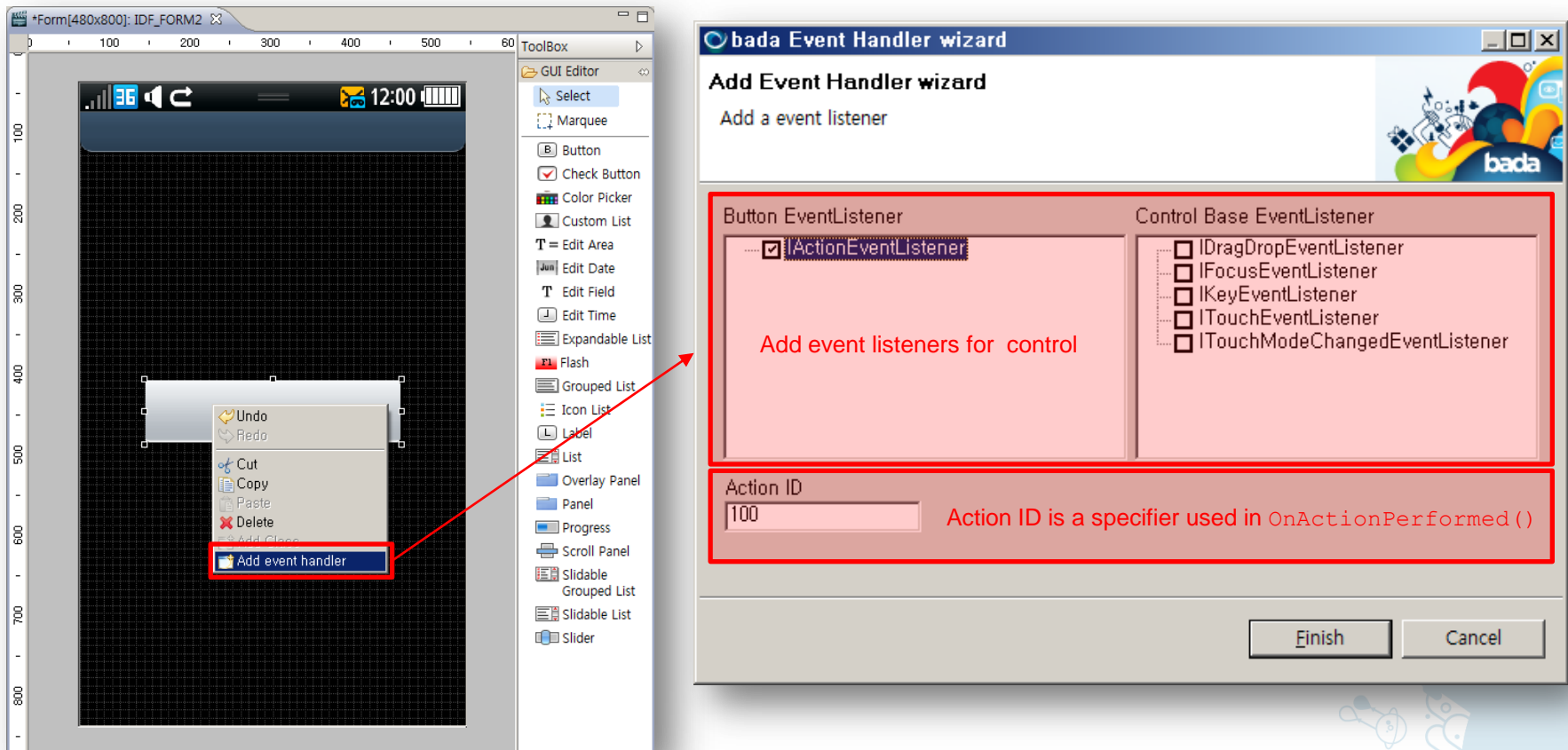
1. Enter a class name for the form.
2. Enter names for header and source files.



Creating Form Classes for the Designed Forms (2/2)

You can add event listeners for controls in the created form class:

1. Select event listeners.
2. Enter an action ID when a specific event handler for each control is selected. The ID must be unique within a form.



Constructing Forms

- The files for the new form are shown in the **Project Explorer**.
 - The source file is `<class name>.cpp` and the header file is `<class name>.h`.
- You can construct the form and add it to the frame as shown in the following source code:
 - The name of the form class is `Form1`. It is constructed in `MyApp::Initialize()`.

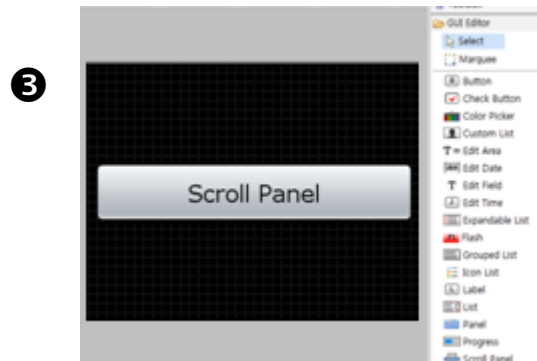
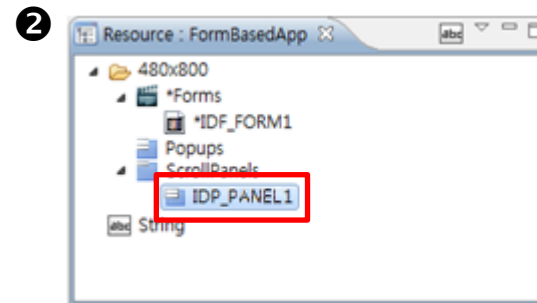
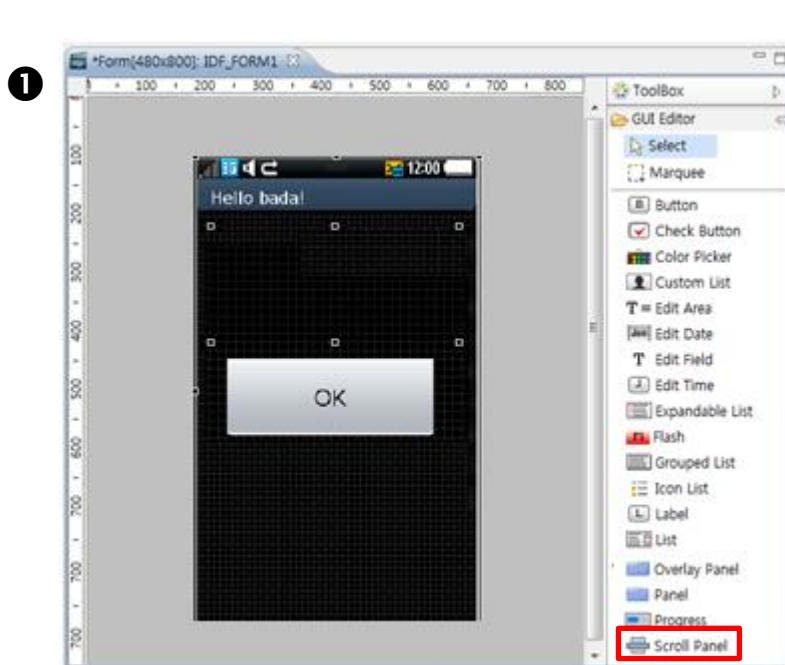
```
#include "Form1.h"

MyApp::Initialize(...)
{
    ...
    Form1 *pForm1 = new Form1();
    pForm1->Initialize();
    pFrame->AddControl(*pForm1);
    pFrame->SetCurrentForm(*pForm1);
    pFrame->RequestRedraw();
    ...
}
```

Using Scroll Panels

You can add scroll panels to forms and design the panels in the same way as forms:

- 1 Add the scroll panel to a form.
- 2 Double-click the scroll panel to open it in the designer.
- 3 Design the scroll panel.

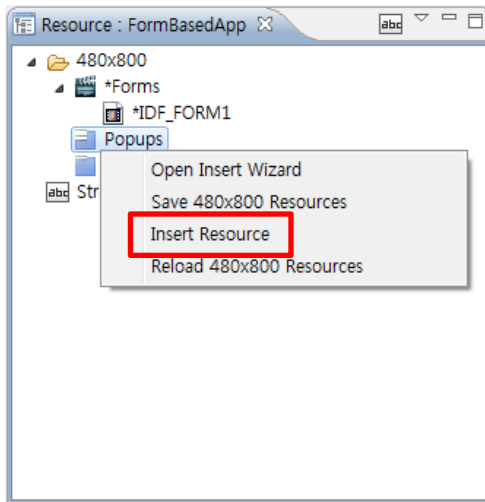


Using Pop-ups

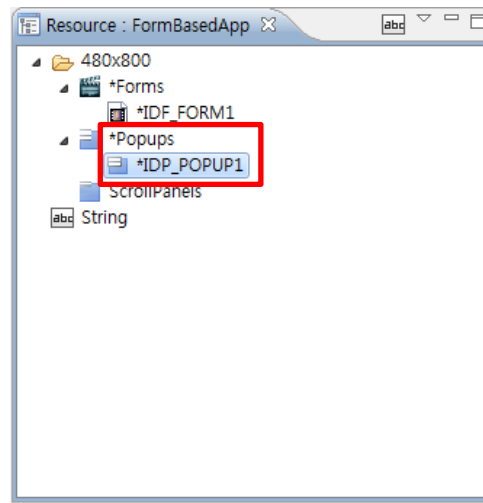
You can add and design pop-ups in the same way as forms:

- ➊ Add the pop-up resource to the project.
- ➋ Double-click the pop-up to open it in the designer.
- ➌ Design the pop-up.

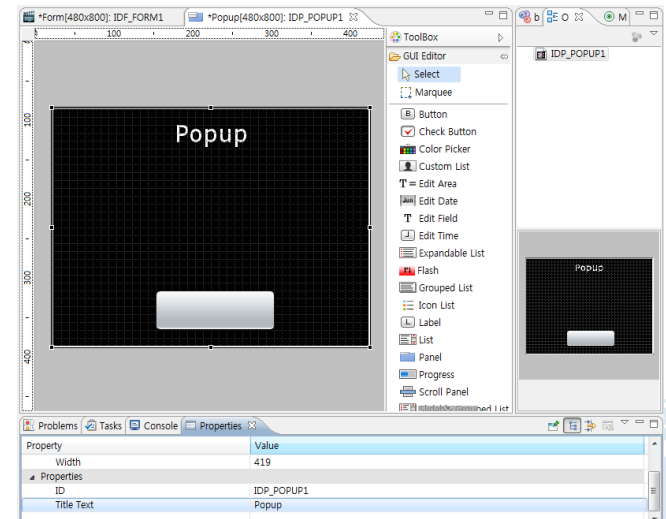
➊



➋



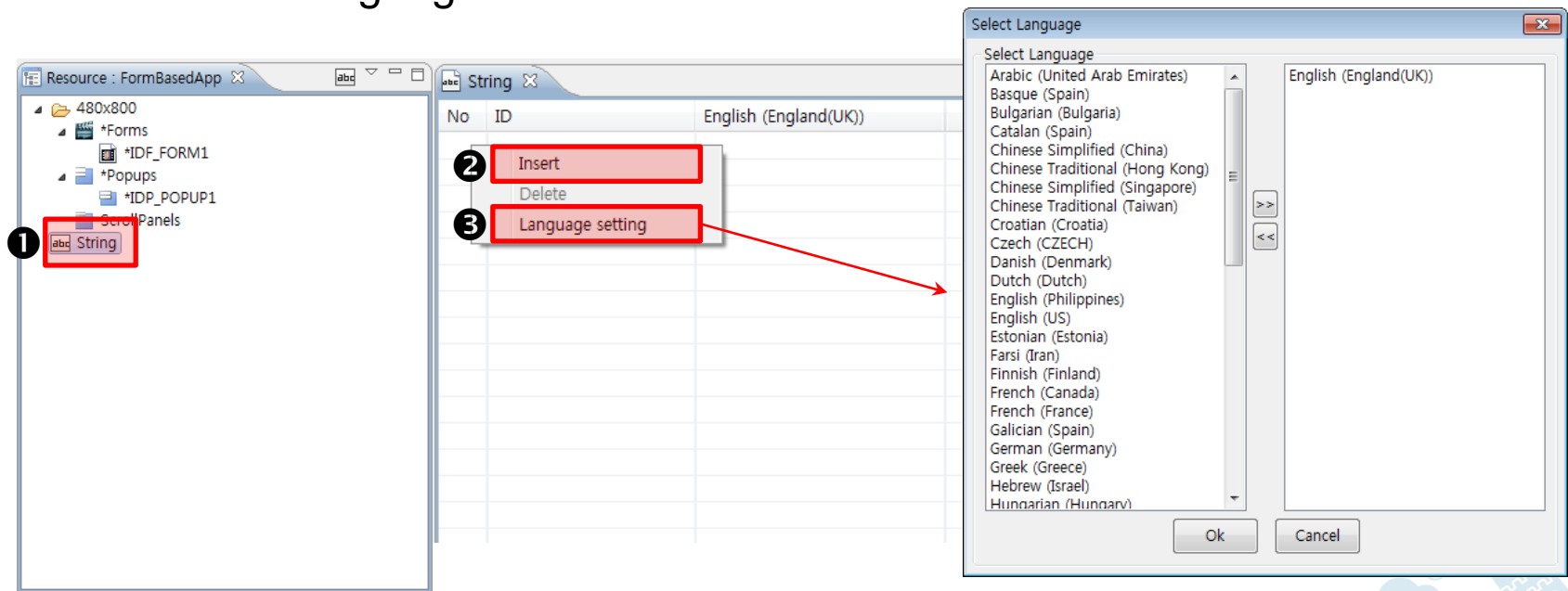
➌



Using String Tables (1/2)

You can use string tables for easy resource management and localization.

- ❶ Double-click the string table.
- ❷ Add new strings.
- ❸ Add new languages.



Using String Tables (2/2)

- Strings in string tables can be used in source code.

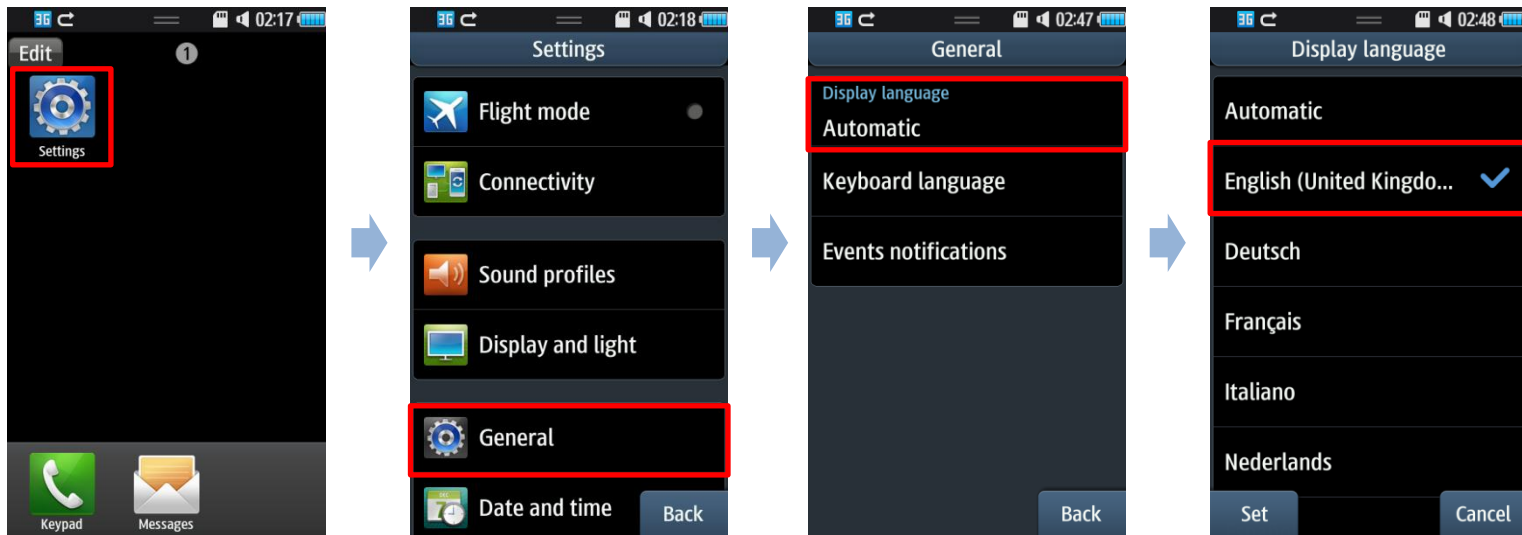
```
...  
Application* pApp = Application::GetInstance();  
String str1;  
r = pApp->GetAppResource()->GetString("IDS_STRING1", str1);  
...
```

- If a string table does not exist for a language, English is used. If the string table does not exist even for English, `GetString()` fails.
- To make the updated string tables active on the device, go to **Settings > General > Display language** and select the corresponding language.

Setting Display Language

You can define the display language under **Settings**.

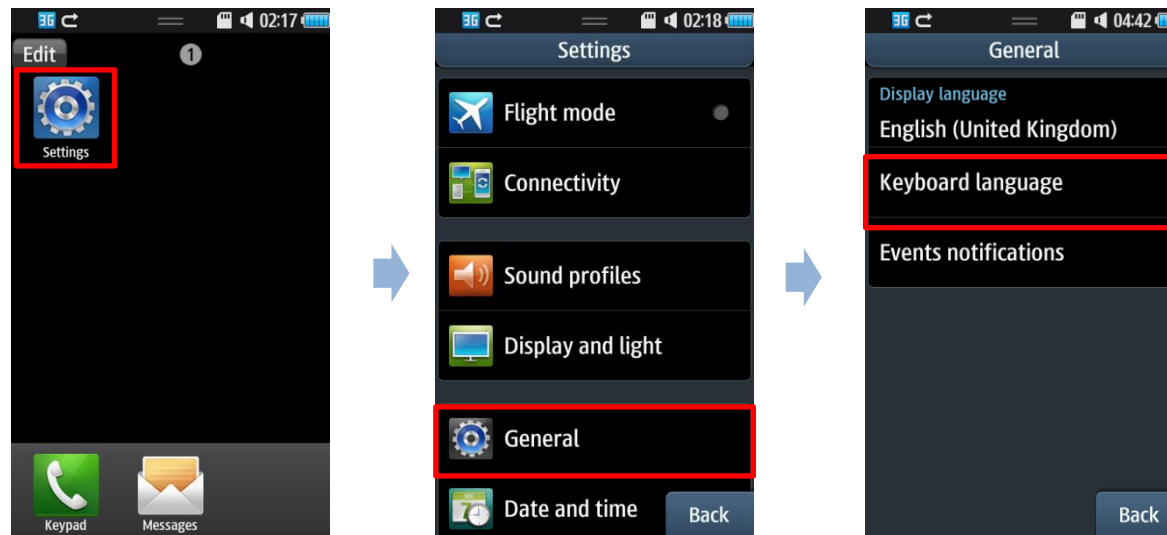
1. Go to **Settings > General > Display language**.
2. Select the language to be used.



Adding New Keyboard Languages (1/3)

You can add a new keyboard language under **Settings**.

1. Go to **Settings > General > Keyboard language**.



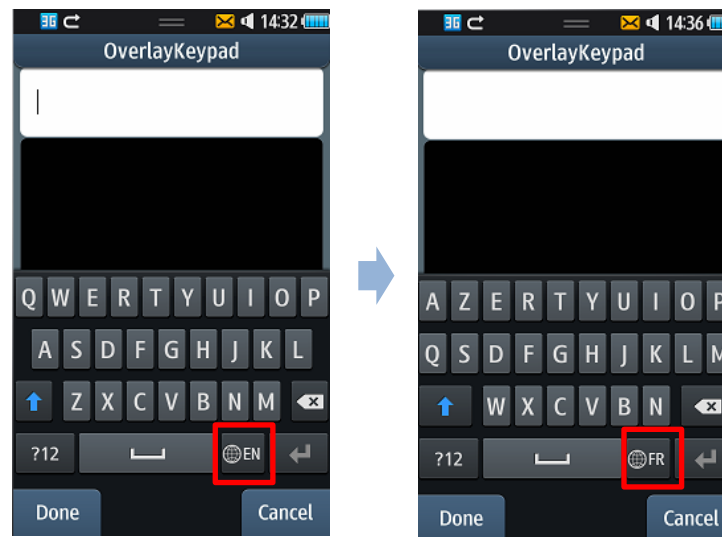
Adding New Keyboard Languages (2/3)

2. Select the new languages.



Adding New Keyboard Languages (3/3)

After the new languages are added, you can press the language button on the virtual keyboard to toggle the language selection.



bada Testing Tool



Contents

- Overview
- Testing Applications
 - Creating a Test Project
 - Creating Test Cases
 - Running a Test Project
- Collecting Test Coverage Data



Overview

- The bada Testing Tool allows you to test your application on a unit level.
- bada Testing Tool contains the following functionality:
 - Wizards
 - Make unit testing easier and faster.
 - Configure a unit testing environment.
 - Generate method stubs for methods in your application that you want to test.
 - Code coverage tool
 - The tool calculates the percentage of application code that is covered by the test cases.
 - You can view the generated coverage data in the **Coverage** view.
 - Based on the coverage data, you can modify the test cases to improve the software quality.
 - bada testing framework
 - The testing framework provides an API that enables you to develop different types of test cases. For more information on the testing framework, see the bada Developer Guide.

Testing Applications

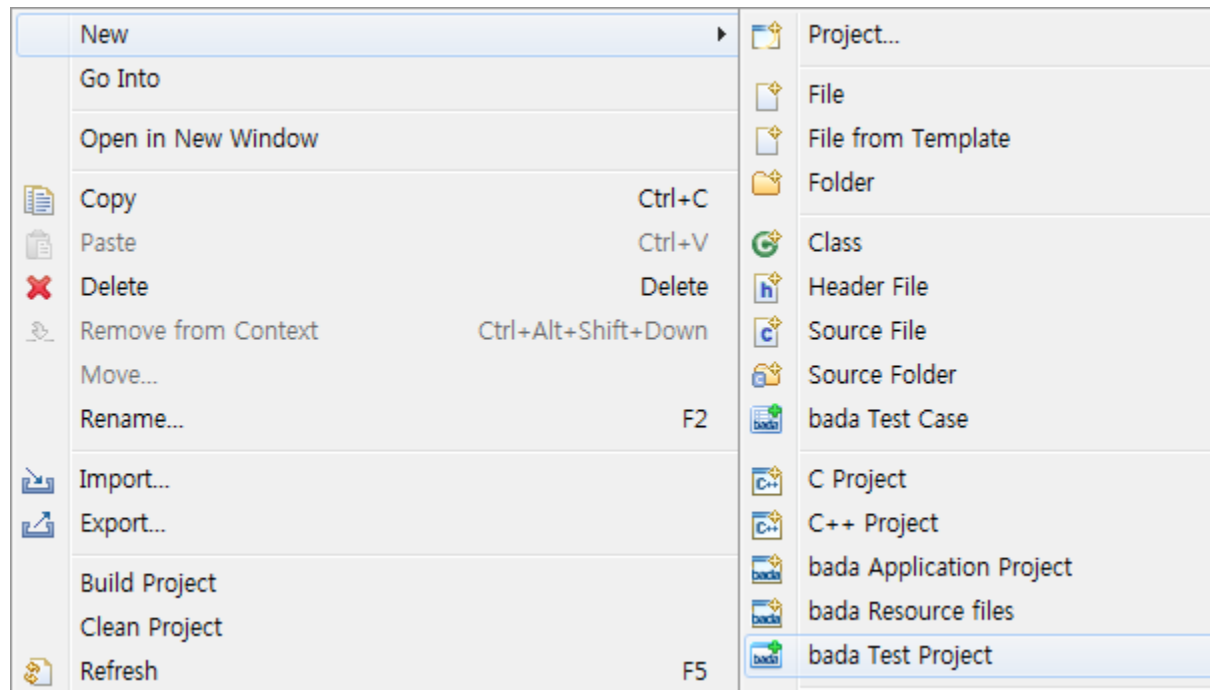
- The bada Testing Tool contains wizards that enable you to build an automated test framework and generate method stubs in a quick and easy manner.
- To test your application:
 1. Create a test project with the Test Project Wizard.
 2. Create the test cases with the Test Case Wizard.
 3. Run the test project on the Simulator or a target device.



Creating a Test Project (1/3)

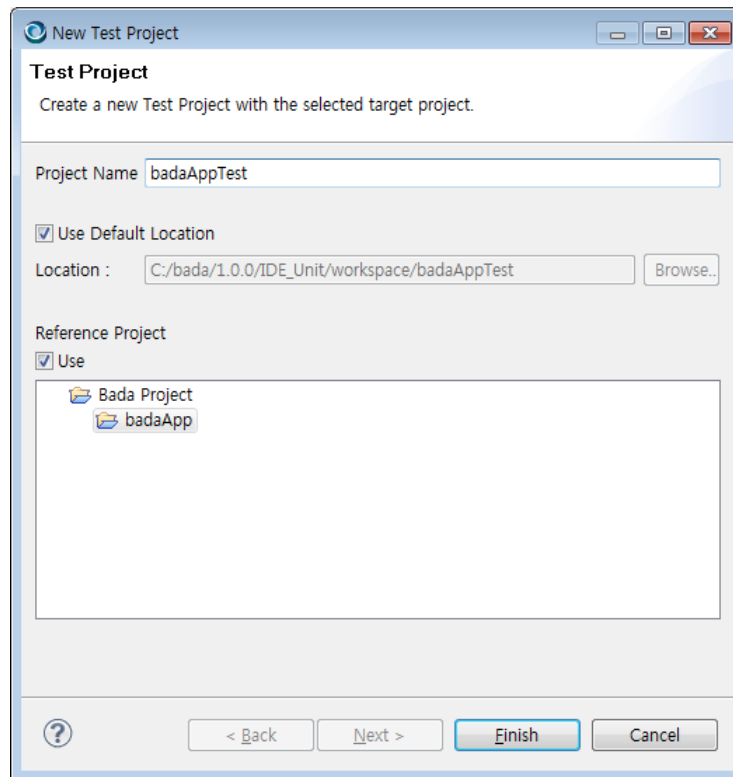
To create a test project:

1. In the **Project Explorer** view, right-click the application project that you want to test and select **New > bada Test Project**.



Creating a Test Project (2/3)

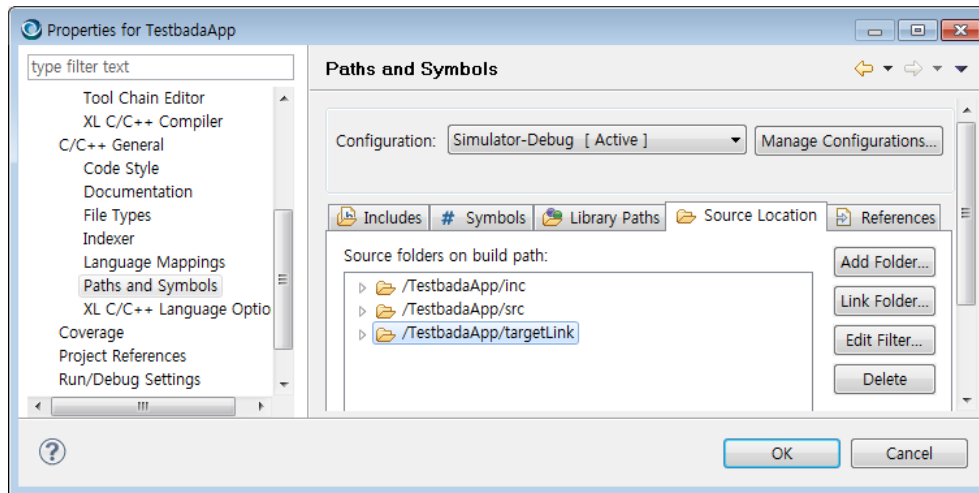
2. In the **New Test Project** window:
 - Give a name and destination folder for the test project
 - Select the **Use** check box and then select the existing bada project you want to test from the list of available reference projects.



3. Click **Finish**.

Creating a Test Project (3/3)

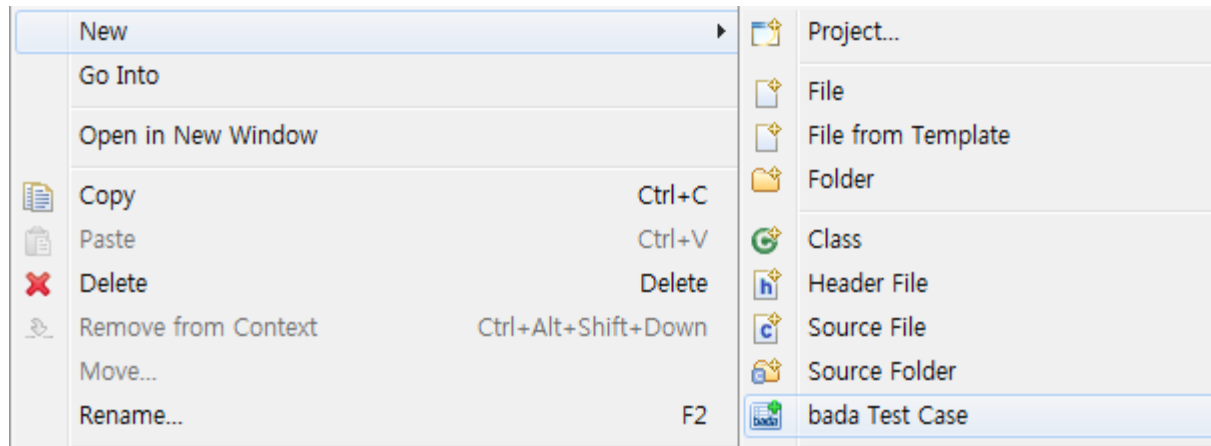
- The generated test project contains a `targetLink` folder that contains the files and resources of the target application project to be tested.
 - The `targetLink` folder is automatically added to the build path in the test project properties along with the `inc` and `src` folders.



- The test project also uses a copy of the application project's `manifest.xml` file.

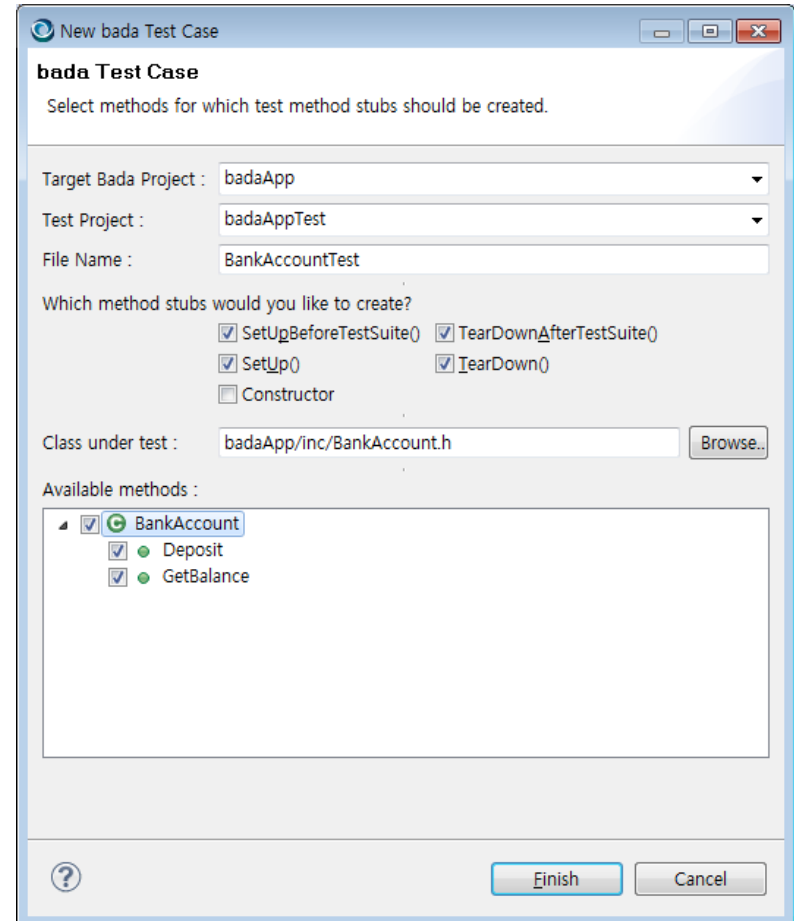
Creating Test Cases (1/3)

- After you have created the test project, you must create the required test cases and add them to the test project.
- To create a test case:
 1. Right-click a header file or `.cpp` file in the application project that you want to test and select **New > bada Test Case**.



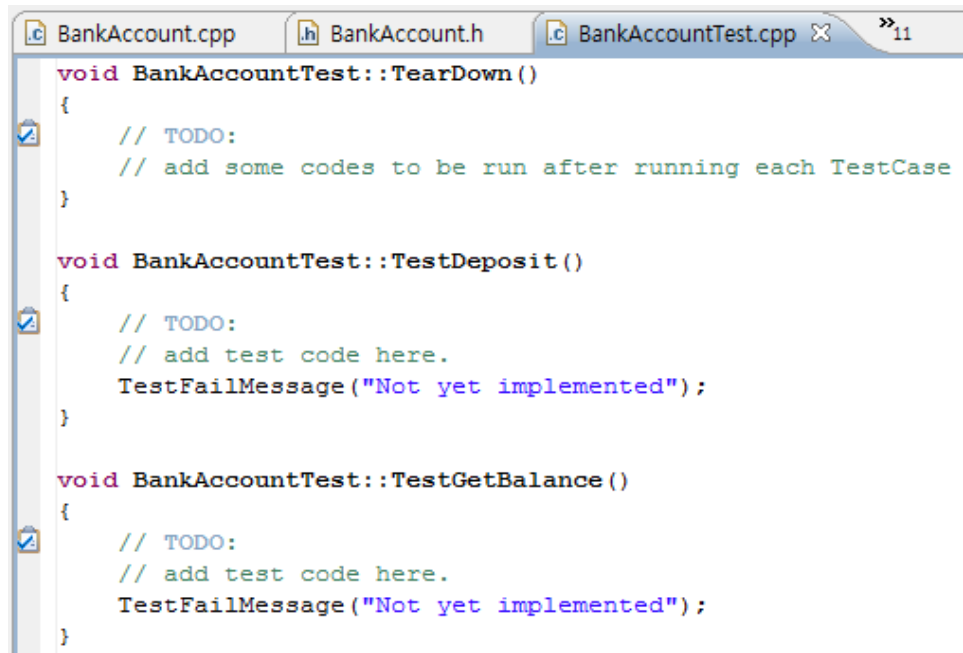
Creating Test Cases (2/3)

2. In the **New bada Test Case** window, define the details for the test case:
 - Make sure that the target bada application project and the test project to which the new test case belongs are correct.
 - Give a name for the test case.
 - Select the method stubs that you want to create and the class methods that you want to test.



Creating Test Cases (3/3)

3. Click **Finish**.
 - The wizard updates the `targetLink` folder and generates method stubs.
4. Implement the actual content of the test case.
 - The auto-generated method stubs are empty and they do not have any code implementation. These method stubs provide a default assertion that displays a “Not yet implemented” failure message.
 - You must implement the method stubs selected for the test case to test the application code behavior.



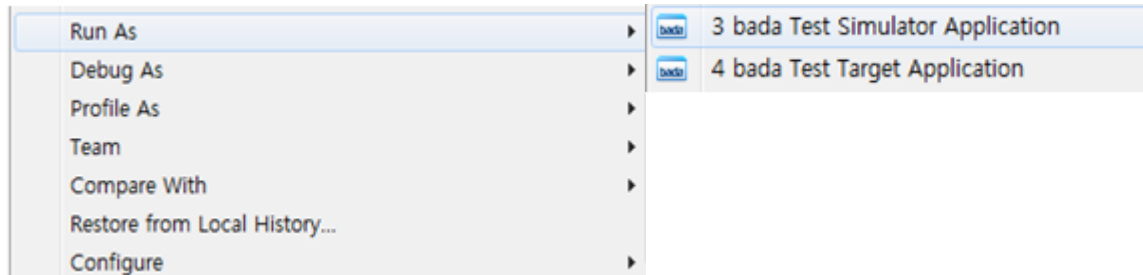
```
BankAccount.cpp  BankAccount.h  BankAccountTest.cpp  »11
void BankAccountTest::TearDown()
{
    // TODO:
    // add some codes to be run after running each TestCase
}

void BankAccountTest::TestDeposit()
{
    // TODO:
    // add test code here.
    TestFailMessage("Not yet implemented");
}

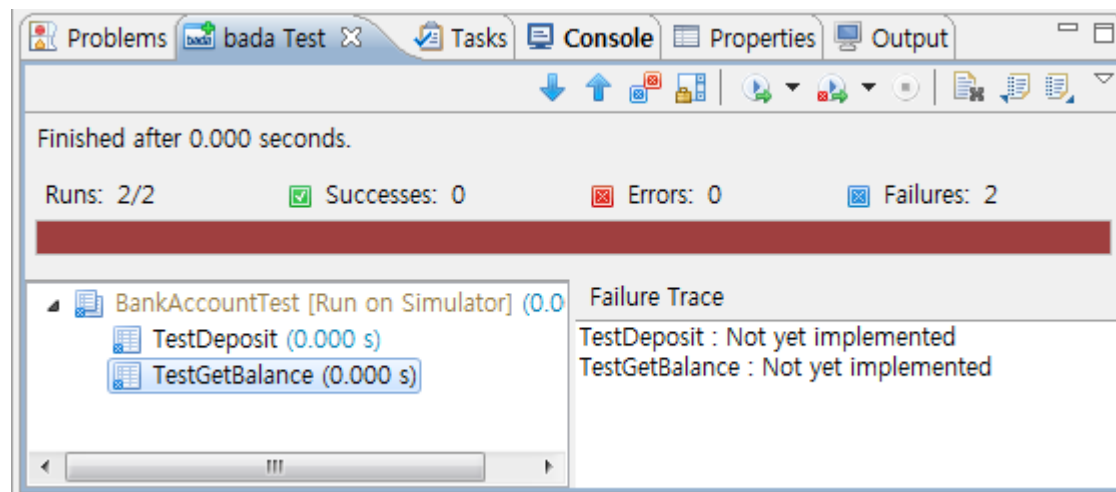
void BankAccountTest::TestGetBalance()
{
    // TODO:
    // add test code here.
    TestFailMessage("Not yet implemented");
}
```

Running a Test Project (1/2)

- You can run your test project on the Simulator or a target device.
- To run a test project, right-click the test project and select **Run As** and then the appropriate testing environment:
















- The test results are displayed in the **bada Test** view.



Running a Test Project (2/2)

You can use the following function buttons in the **bada Test** view:

-  Move to the next failed test case
-  Move to the previous failed test case
-  Show failures only in the test result view
-  Lock the **bada Test** view scroll bar
-  Rerun all tests
-  Rerun failed and erroneous test cases only
-  Terminate the test execution
-  Clear the test results
-  Import the XML file that contains the test results
-  Export the test results to an XML file
-  Test cases executed with assertion failures
-  Test cases successfully executed
-  Test cases executed with exceptions

Collecting Test Coverage Data (1/5)

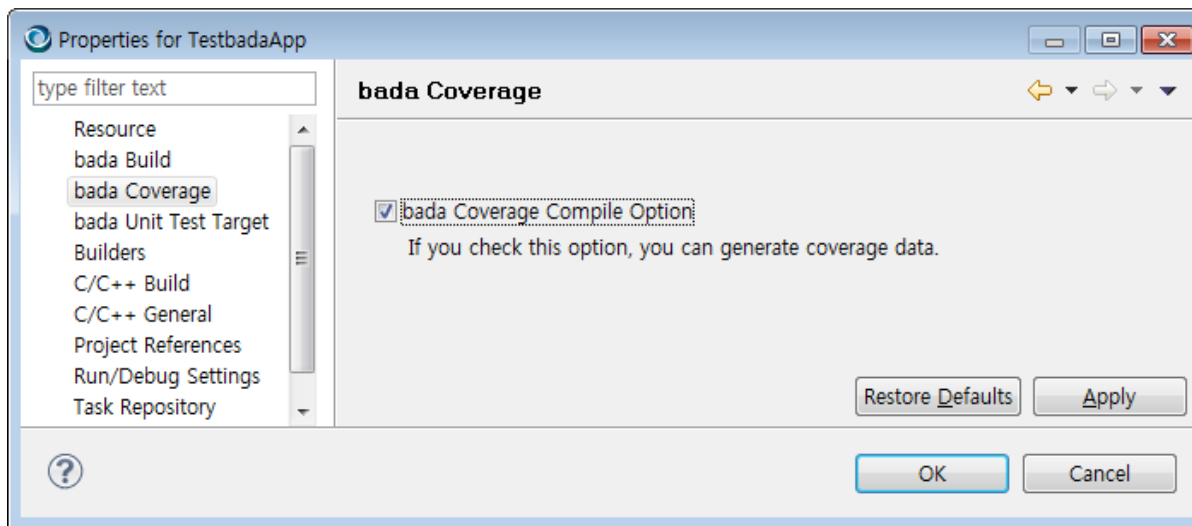
- You can use the code coverage tool to check how comprehensively your application has been tested.
- The code coverage tool uses a wide variety of criteria to calculate the coverage. It supports line coverage, indicating the percentage of lines of code that is tested.
- The code coverage tool supports the Simulator only.



Collecting Test Coverage Data (2/5)

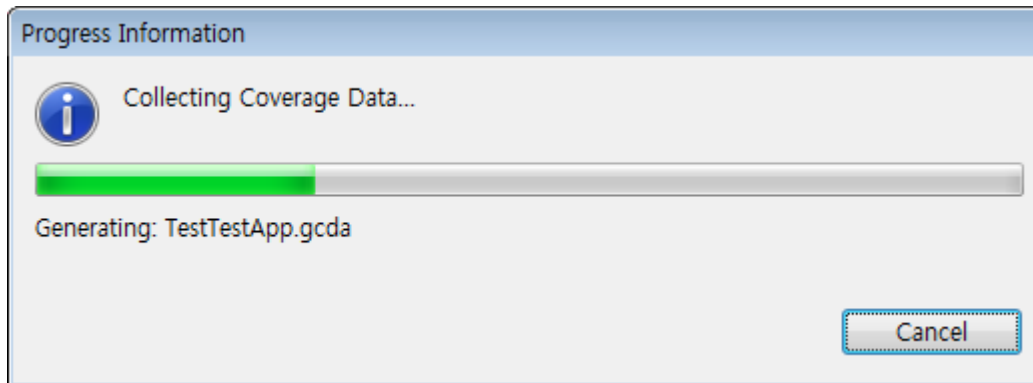
To be able to view the coverage data for your application during unit testing:

1. In the **Project Explorer** view, right-click the application project under testing and select the coverage compilation option in **Properties > bada Coverage**.
2. Click **OK**.
3. Run your test project.
4. Close the Simulator.



Collecting Test Coverage Data (3/5)

5. Click the **Coverage** button in the toolbar to run the code coverage tool and open the **Coverage** view.



Collecting Test Coverage Data (4/5)

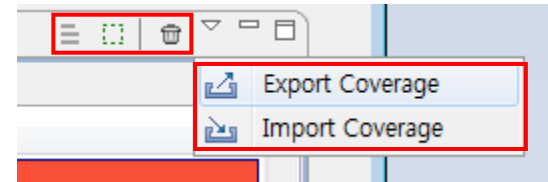
- The lines of code covered in the test are highlighted in blue, and the lines of code that are not covered are highlighted in red.
- You can view the calculated coverage data (percentage of covered and uncovered lines of code) in the **Coverage** view.

The screenshot displays an IDE with three source files open: BankAccountEntry.cpp, BankAccount.cpp, and Bank.cpp. The code in BankAccount.cpp is highlighted in blue (covered) and red (not covered). The Coverage view at the bottom shows the following data:

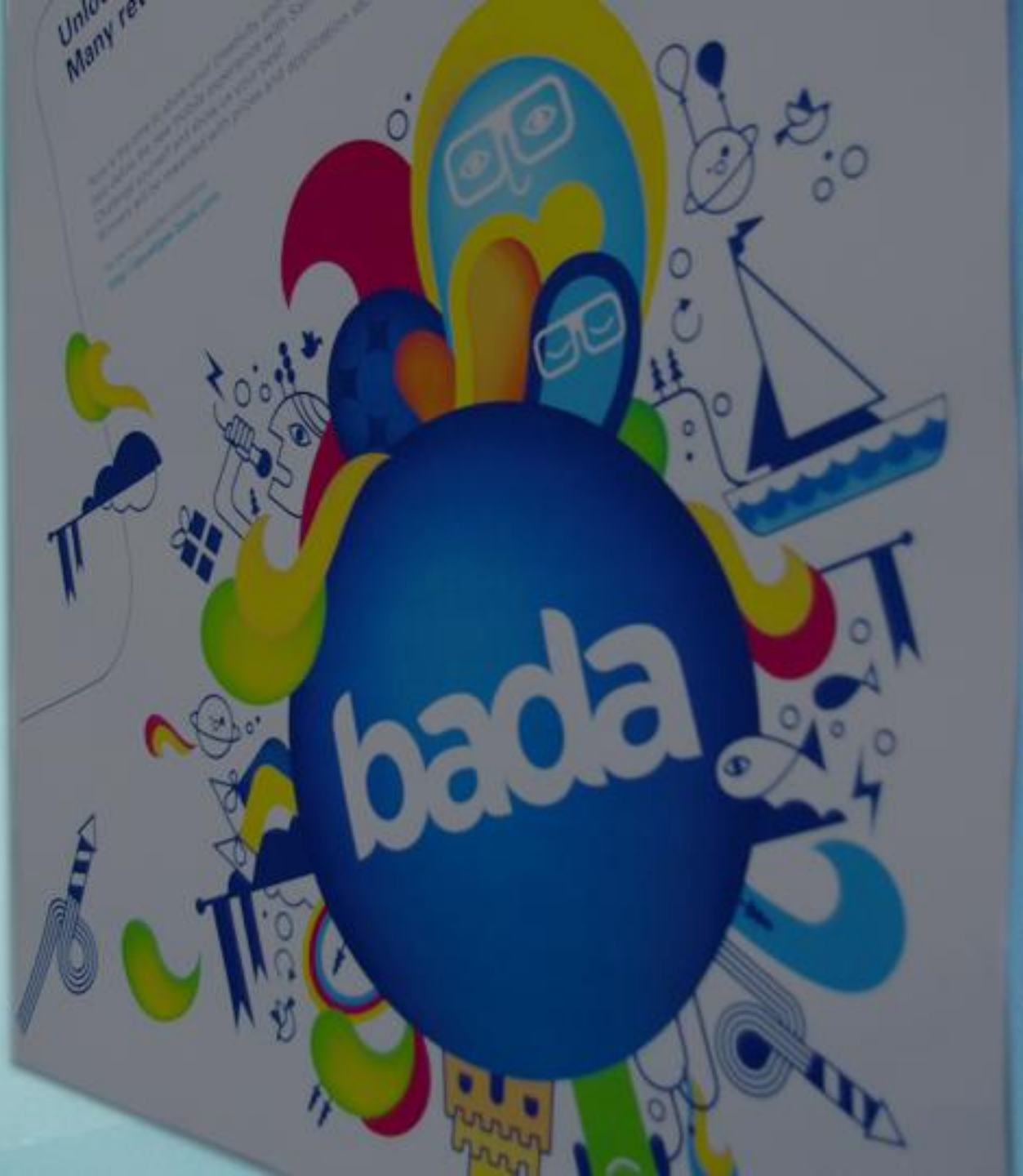
File	Lines
src	
BankAccountEntry.cpp	87.50%
BankAccount.cpp	68.97%
Bank.cpp	0.00%
Total	69.39%

Collecting Test Coverage Data (5/5)

- You can manage the coverage data using the function buttons in the **Coverage** view.
 - 🗑 Clean coverage data
 - 👁 Hide coverage lines in source files
 - ☰ Show coverage data
 - 📄 Export coverage data
 - 📄 Import coverage data



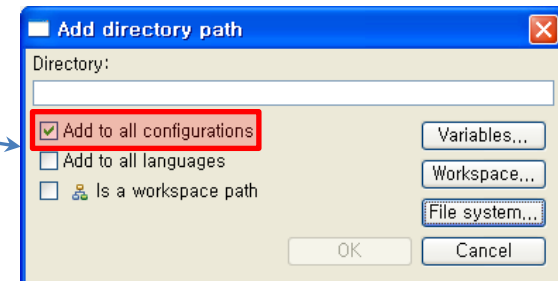
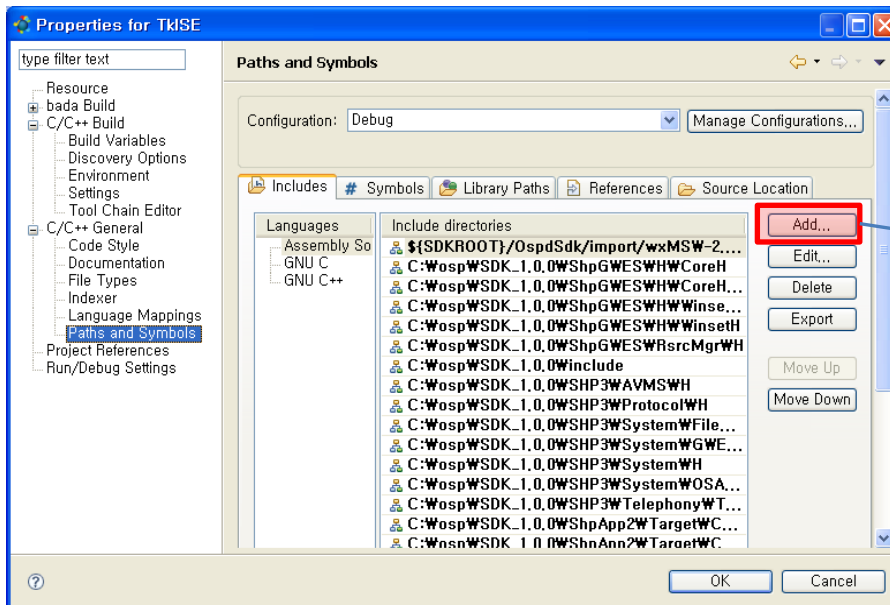
FAQ



FAQ (1/8)

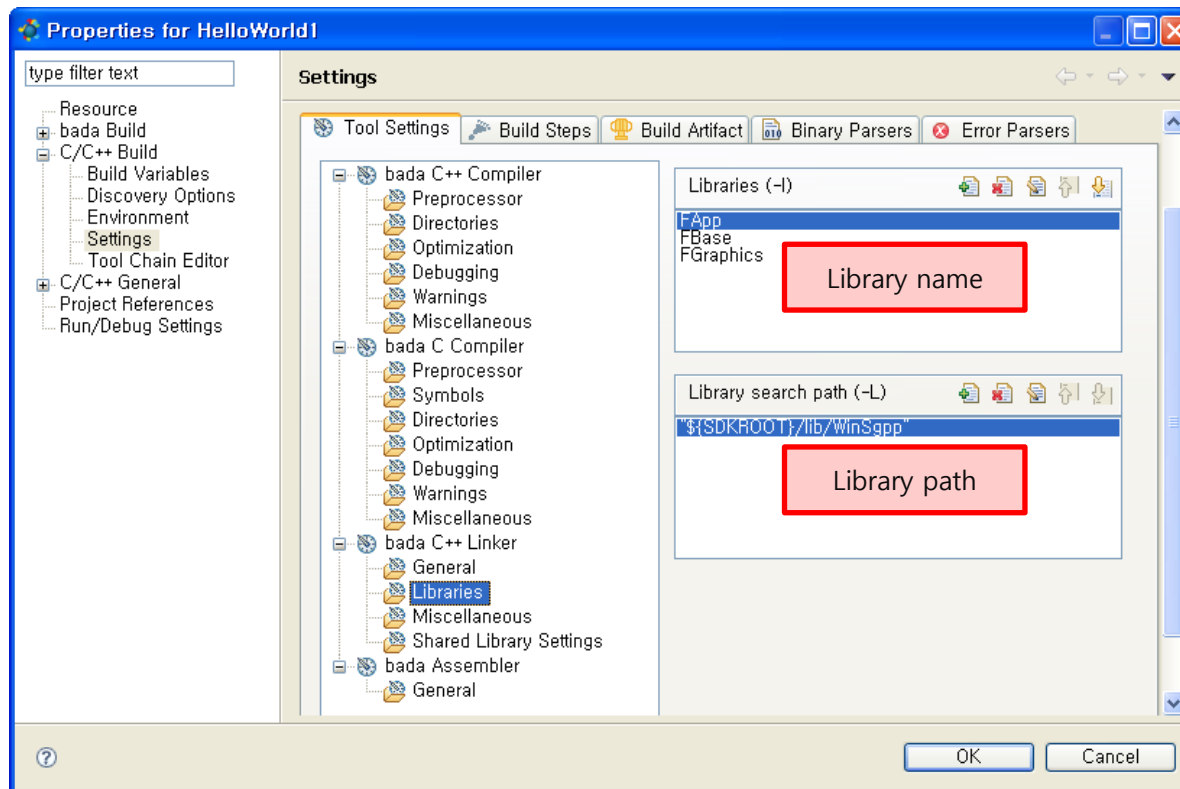
How do I convert Visual Studio projects into bada application projects?

1. Create a new bada application project with bada IDE.
2. Copy the source and resource files to the created bada project folder to the appropriate subfolders (source files to `/src`, include files to `/inc`, and resource files to `/Res`).
3. Set the compilation flags:
 - Include paths in: **Project Properties > C/C++ General > Paths and Symbols**



FAQ (2/8)

4. Define linker options in: **Project Properties > C/C++ Build > Settings > bada C++ Linker**



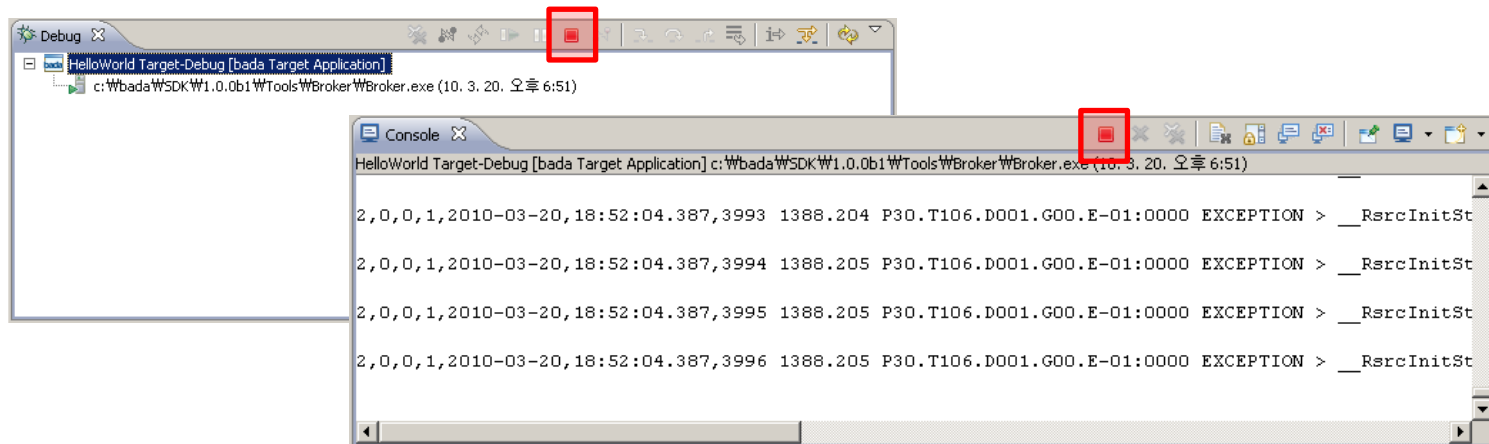
FAQ (3/8)

- How do I improve the execution performance of the bada IDE?
 - You can tweak the memory size to fit your development machine.
 - Change the `badaIDE.ini` configuration file in the bada IDE folder:
 - `Xms40m` → `Xms256m`
 - `Xmx256m` → `Xmx1024m`
- How do I cope with the occurrence of *java.lang.OutOfMemoryError: PermGen space* errors in the bada IDE?
 - PermGen means the permanent generation of objects in the VM. Fix the error by appending the following to the `badaIDE.ini` configuration file in the bada IDE folder:
 - `XX:PermSize=64m`
 - `XX:MaxPermSize=128m`



FAQ (4/8)

- How do I cope with the occurrence of the build error “cs-make: *** [src/source.o] Error -1073741819”?
 - The error code 1073741819 indicates an “Error Access Violation”, which means that the compiler has run out of memory.
 - Rebuild the application when you encounter this error.
- How do I cope with errors that occur when running or debugging an application on the target device?
 - In the bada IDE, click the **Terminate** button in the **Console** or **Debug** view to terminate the run or debug process.



FAQ (5/8)

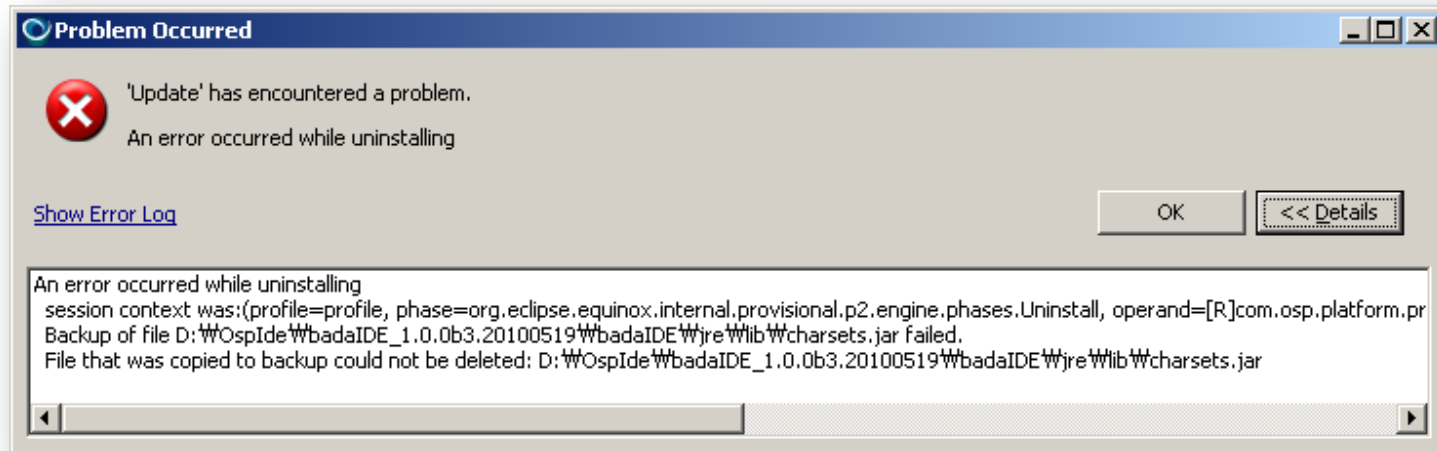
How do I cope with a situation in which the declaration cannot be found or is found in the wrong include files when executing the “open declaration” command? This problem occurs when I execute the command after modifying the bada SDK root or after importing a bada Application project.

- Right-click the project and select **Index > Rebuild**.



FAQ (6/8)

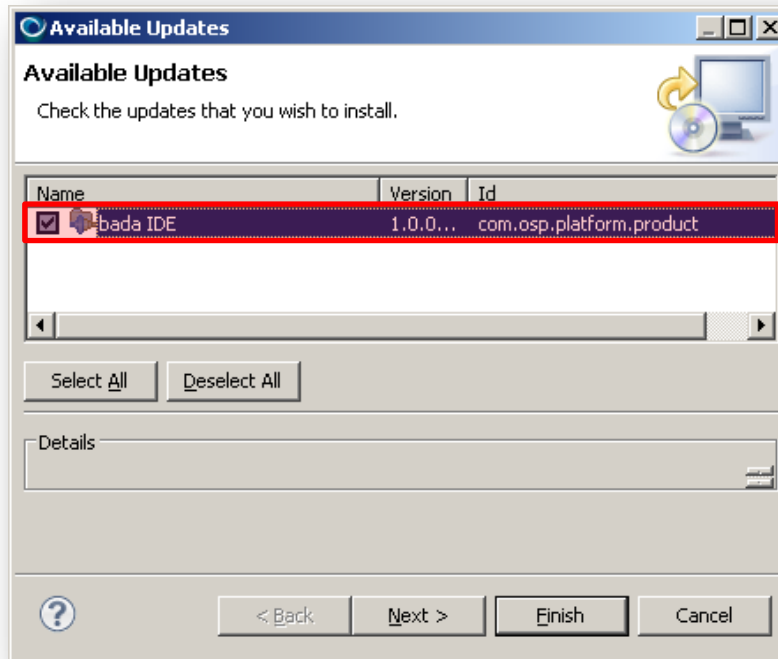
How do I cope with errors, such as the one shown below, that occur when updating the bada IDE?



1. Exit the bada IDE.
2. Do one of the following:
 - If you have already installed JRE or JDK, remove the `<BADA_SDK_HOME>\IDE\jre` directory.
 - If you have not installed JRE or JDK, move the `jre` directory from `<BADA_SDK_HOME>\IDE` to `<BADA_SDK_HOME>`.
Add `<BADA_SDK_HOME>\jre\bin` to the **PATH** environment variable.
3. Start the bada IDE.

FAQ (7/8)

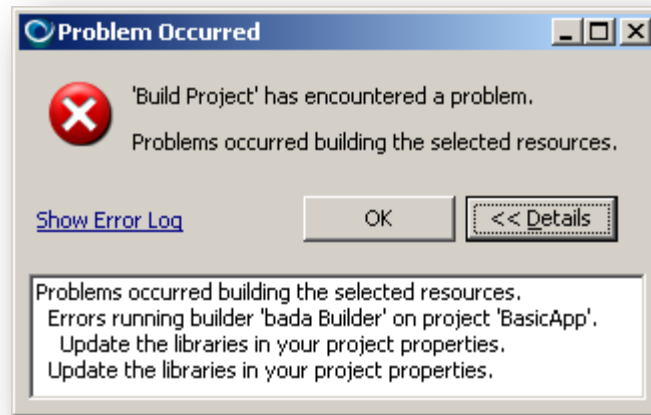
4. In the bada IDE menu, go to **Help > Check for Updates**.
5. If an update for the bada IDE is found, select it and click **Next** to continue and complete the installation process.



6. If you moved the `jre` directory from `<BADA_SDK_HOME>\IDE` to `<BADA_SDK_HOME>` before installing the updates, move it back to `<BADA_SDK_HOME>\IDE`. Remove `<BADA_SDK_HOME>\jre\bin` from the `PATH` environment variable.

FAQ (8/8)

How do I cope with errors, such as the one shown below, that occur when building a bada project?



1. Exit the bada IDE.
2. Copy `fosp_update.exe` from `<BADA_SDK_HOME>\IDE\` to the root directory of the project.
3. Run `fosp_update.exe` in the project root directory.
4. Start the bada IDE and rebuild the project.



bada

The platform with more opportunities
Invitation to Adventure