Tizen SDK Development Guide

Table of Contents

Introduction

Tizen SDK is composed of many separate packages which have their own dependencies. And DIBS(Distributed Intelligent Build System) is the build system designed for building the kind of complexity needed. It provides various features.

- Has own packaging system and packaging interface

- Provides distributed package server/build Server
- Provides Automatic dependency checker/resolver
- Provides client/server tools which are easy to use

This guide will show how to build Tizen SDK packages using DIBS

Setup Build Environment

Linux(Ubuntu)

Install Tizen SDK

- Install the SDK with “SDK Development Tools”

Install Ruby

- To use DIBS, you have to install Ruby 1.8.7

```
sudo apt-get install ruby
```

- Higher version of Ruby is not tested yet!

Install Packages needed by DIBS

```
sudo apt-get install wget zip unzip
```

Install Pre-Requisite Packages

To build or develop SDK, you have to install the following packages

- For emulator development:
  - bcc bison flex autoconf gcc libglu1-mesa-dev libsdl1.2-dev libgtk2.0-dev libsdl-image1.2-dev libsdl-gfx1.2-dev debhelper libxml2-dev libasound2-dev libx11-dev zlib1g-dev uuid-dev libv4l-dev
- GDB 7.2
  - quilt libncurses5-dev libexpat1-dev libreadline-dev mingw32(only for building windows version)

- GCC 4.5
  - quilt texinfo bison flex mingw32(only for building windows version)

```
sudo apt-get install {pre-requisite packages}
```

More

For your convenience,

- Add DIBS path to $PATH in shell configuration file. ex) .bashrc

```
export PATH={SDK Install dir}/tools/dibs:$PATH
```

Windows

Install Tizen SDK

Install the SDK first

Install Ruby

- You can download Ruby binary at..

```
ftp://ftp.ruby-lang.org/pub/ruby/binaries/mswin32/ruby-1.8.7-i386-mswin32.zip
http://rubyforge.org/frs/download.php/75679/rubyinstaller-1.8.7-p357.exe
```

Install rubyzip module

- You should install ruby zip module after installing ruby

```
c:\Ruby187\bin> gem.bat install rubyzip
```

Install Packages needed by DIBS

- Login MinGW provided by Tizen SDK. Execute following windows BAT file.

```
{ SDK Install dir}/tools/mingw/msys/1.0/msys.bat
```

- Execute the following commands on MinGW environment

```
$ mingw-get.exe update
$ mingw-get.exe install msys-wget
$ mingw-get.exe install msys-zip
$ mingw-get.exe install msys-unzip
```

Install MSYS GIT

- Download the MSYS binary

```
http://msysgit.googlecode.com/files/Git-1.7.9-preview20120201.exe
```

- Install it

More

For your convenience,

- Add "/usr/bin/ruby" shell script

```
#!/bin/sh
{ruby install dir}/bin/ruby.exe $@
```

ex)

```
#!/bin/sh
/c/Ruby187/bin/ruby.exe $@
```

- Add "/usr/bin/git" shell script

```
#!/bin/sh
{MSYS GIT install dir}/bin/git.exe $@
```

ex)

```
#!/bin/sh
/c/Program\ Files/Git/bin/git.exe $@
```

- Add DIBS path to $PATH in shell configuration file. ex) /etc/profile

```
export PATH={SDK Install dir}/tools/dibs:$PATH
```

Simple Local Build and Test

If you have downloaded an SDK source package and modified it, and you would want to build and apply it to the Tizen SDK, Please refer the following process.

Build Package

Building a SDK package is very simple. Here is the command for building the package.

```
## pkg-build -u <package server url> [-o <os>] [-c] [-h] [-v]
## -u : Package server URL which contains binary and development packages.
        If ommited, it will use previous server URL.
## -o : Target OS(ubuntu-32/ubuntu-64/windows-32/windows-64/macos-64)
## -c : Clean build""
        If set, start build after downloading all dependent packages
        If not set, it will not download dependent packages if already downloaded
## -h : Display help
## -v : Display version
```

And Here are simple steps

1. Git clone and move to source directory
    - ex) **$> git clone review.tizen.org:sdk/ide/common-eplugin**
    - ex) **$> cd common-eplugin**
2. Type the command
    - ex) **$> pkg-build -u http://<package_server_address>/<package_server_name>/<distribution_name>**
3. Now you can see the package files(*.zip, *.tar.gz) in your source directory

Install Package

Installing a SDK package is also very simple. Here is the command for installing package files

```
## pkg-cli install-file -P <package file path> [-l <location>] [-u <package server url>] [--trace] [--force]
## -P : Binary package file(*.zip) path which you want to install
## -l : Install root location of target SDK
```

```
       If omitted, current working directory path will be set
## -u : Package server URL which contains binary and development packages.
##       If ommited, it will use previous server URL.
##       ex) http://172.21.17.55/dibs/unstable
## --trace : Install the package with all dependent packages
## --force : Install the package by force
       This option will allow installing the package that has lower or equal version compare to installed
```

Now let's assume that you have just finished building and have a Tizen SDK installation on '~/tizen-sdk'

1. Just type the command
   - ex) **$> pkg-cli install-file -P common-eplugin_0.20.6_linux.zip -l ~/tizen-sdk**

Launch your own SDK

Now you can check your modifications. Launch your SDK!!

1. Type the following command or use the short-cut for launching Tizen SDK.
   - ex) **$> ~/tizen-sdk/ide/startup.sh**

Simple Remote Build and Test

If you want to modify the Tizen SDK source file and upload it to the package server using build server then using DIBS,here is the simple process to do it.

Download source code and modified and push to upstream

- Tizen control source code using git.
  1. Download Tizen source code using git command
  2. Modified source code
  3. Push to upstream using git command
     - If source code change is accepted then you can build using build server
  4. Execute remote build command
     - If source code builds successfully then upload the package file to the package server.

Remote build command

Remote build command is simple.

```
## build-cli build -N <project name> -d <server address> [-o <os>] [-w <password>] [--async]'
## -N : Project name. This should be set before through "build-svr add-prj" command
## -d : Build server address: 127.0.0.1:2224
## -o : Target operating system: ubuntu-32/ubuntu-64/windows-32/windows-64/macos-64
## -w : Password for managing project. If a password is set before through "build-svr add-prj" command, you should input the password
## --async : asynchronous job
```

- You can request to build project to build server. After that, package will be uploaded to package server
- The project name should be set before through "build-svr add-prj" command

- Step

1. Request to build project to build server
    - ex) $> build-cli build -N dibs -d <build_server_address>:<build_server_port_number>

More DIBS commands

There are more useful commands provided

List Up Available Packages

You can list up available packages of server.

```
## pkg-cli list-rpkg [-o <os>] [-u <package server url>]
## -o : Target OS(ubuntu-32/ubuntu-64/windows-32/windows-64/macos-64)
## -u : Package server URL which contains binary and development packages.
       If ommited, it will use previous server URL.
```

1. List up packages
    - ex) $> **pkg-cli update -u http://<package_server_address>/<package_server_name>/<distribution_name>**
    - ex) $> **pkg-cli list-rpkg**
2. List up packages with updating

- ex) $> **pkg-cli list-rpkg -u http://<package_server_address>/<package_server_name>/<distribution_name>**

You can list up packages of your install directory

```
## pkg-cli list-lpkg [-l <location>]
## -l : Install root location of target SDK
       If omitted, current working directory path will be set
```

1. List up packages
   - ex) $> **pkg-cli list-lpkg -l ~/tizen-sdk**

Update Package List

You should have package list of server in your host before listing, installing and downloading packages. So, if you want to install the latest package, then you should update your package list before installing.

```
## pkg-cli update [-u <package server url>]
## -u : Package server URL which contains binary and development packages.
       If ommited, it will use previous server URL.
```

1. Update package list from server
   - ex) **$> pkg-cli update -u http://<package_server_address>/<package_server_name>/<distribution_name>**
2. Install / download packages from server
   - ex) **$> pkg-cli install -P nativeapp-eplugin -l ~/tizen-sdk**
   - ex) **$> pkg-cli install -P unittest-eplugin -l ~/tizen-sdk**
   - ex) **$> pkg-cli download -P base-ide-product -l ~/downloads**
3. If package is updated on server and you want to use is, you should update your package list. **If you do not set the server url, it will be set previous server URL.**
   - ex) **$> pkg-cli update**
   - ex) **$> pkg-cli install -P nativeapp-eplugin -l ~/tizen-sdk**

Upgrade Packages

You can upgrade your installed packages from server.

```
## pkg-cli upgrade -l <location> -u <package server url>
```

```
## -u : Package server URL which contains binary and development packages.
##      If ommited, it will use previous server URL.
## -l : Install root location of target SDK
##      If omitted, current working directory path will be set
```

1. Check package for upgrading
    - $> **pkg-cli update -u http://<package_server_address>/<package_server_name>/<distribution_name>**
    - $> **pkg-cli check-upgrade -l ~/tizen-sdk**
2. Upgrade packages
    - $> **pkg-cli upgrade -l ~/tizen-sdk**
3. Upgrade packages with updating
    - $> **pkg-cli upgrade -l ~/tizen-sdk -u http://<package_server_address>/<package_server_name>/<distribution_name>**
4. If you want to upgrade specific package, you can upgrade it as installing
    - $> **pkg-cli install -P common-eplugin -l ~/tizen-sdk -u http://<package_server_address>/<package_server_name>/<distribution_name>**

Install SDK

You can also install new SDK using the network install command. Basically this command is used for installing packages by network. But you can also set meta package as package name like TIZEN-IDE, EMULATOR-TOOLS and EMULATOR-IMAGE.

```
## pkg-cli install -P <package name> [-o <os>] [-u <package server url>] [-l <location>] [--trace] [--force]
## -P : Binary package name which you want to install
## -o : target OS: ubuntu-32/ubuntu-64/windows-32/windows-64/macos-64
       If ommited, it will host os.
## -u : Package server URL which contains binary and development packages.
       If ommited, it will use previous server URL.
## -l : Install root location of target SDK
       If omitted, current working directory path will be set
## --trace : Install the package with all dependent packages
## --force : Install the package by force
       This option will allow installing the package that has lower or equal version compare to installed
```

1. Install "TIZEN-IDE" by network to new location("~/tizen-sdk2")
    - ex) **$> pkg-cli install -P TYPICAL -l ~/tizen-sdk2 --trace** \
2. Change Tizen SDK configuration

- ex) **$> echo "TIZEN_SDK_INSTALLED_PATH=/home/{username}/tizen-sdk2" > ~/tizen-sdk-data/tizensdkpath**

3. Launch your SDK!
   - ex) **$> ~/tizen-sdk2/ide/startup.sh**