DAVE™ SDK – Quick Start

Create HelloWorld APP

Learning Outcome

- Learn the typical workflow of DAVE™ APP development

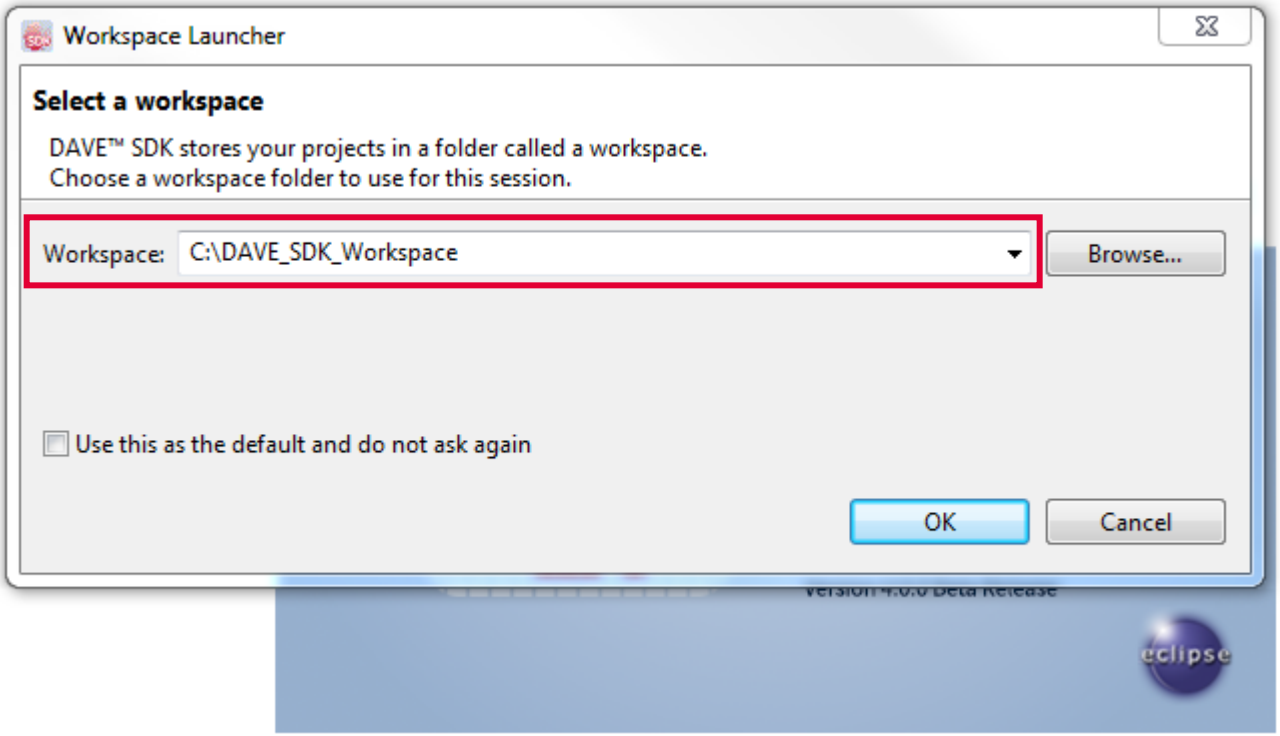
› Learn the basic concepts of DAVE™ SDK for DAVE™ APP development:
  - Create new DAVE™ APP project
  - Design UI
  - Write a template for code generation
  - Update the APP manifest
  - Create update site
  - Import in DAVE™
  - Debug Template

DAVE™ SDK installation

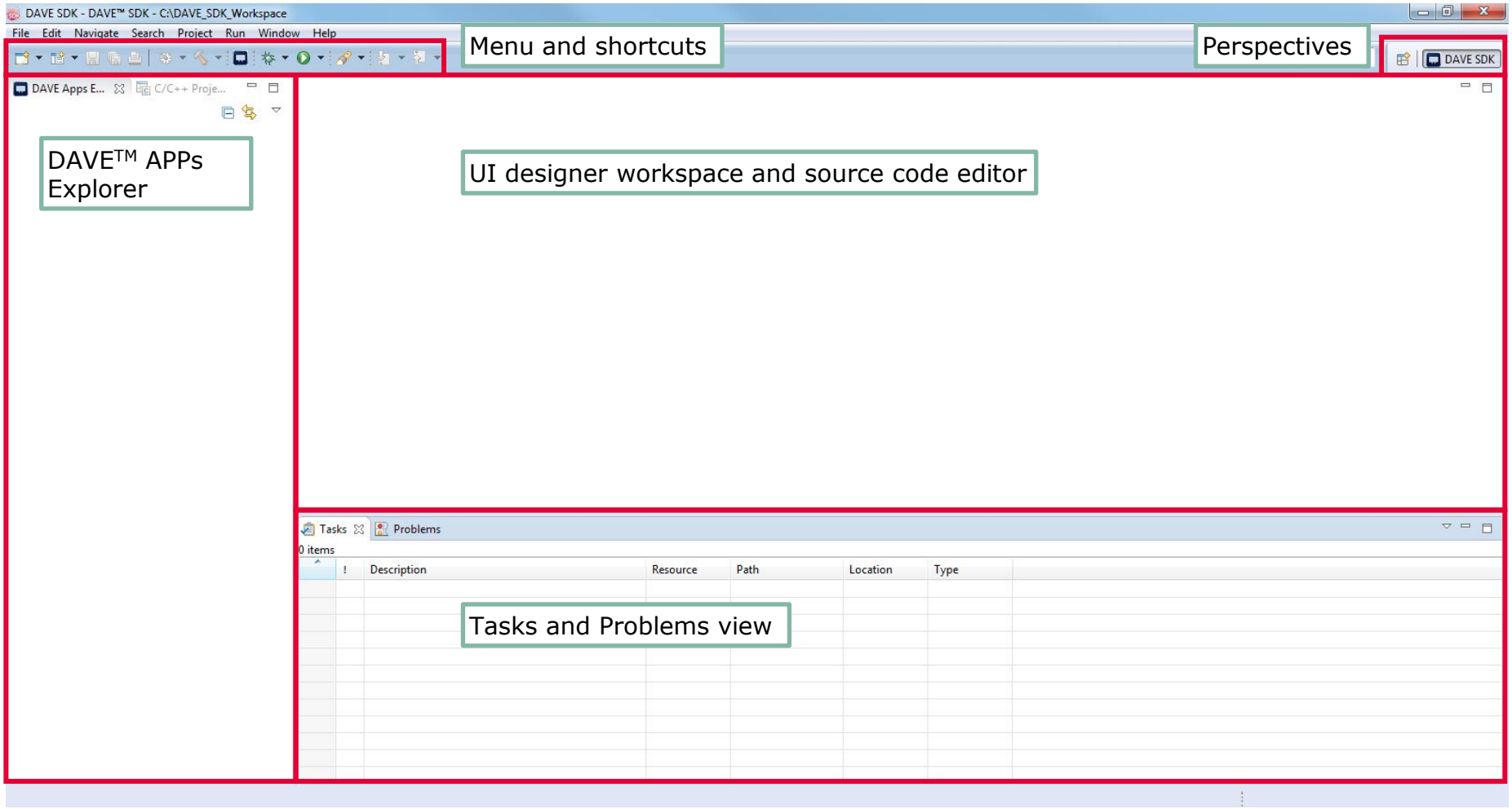
- DAVE™ SDK is part of the DAVE™ download package
- DAVE™ website
  - www.infineon.com/DAVE

Starting DAVE™ SDK for the first time

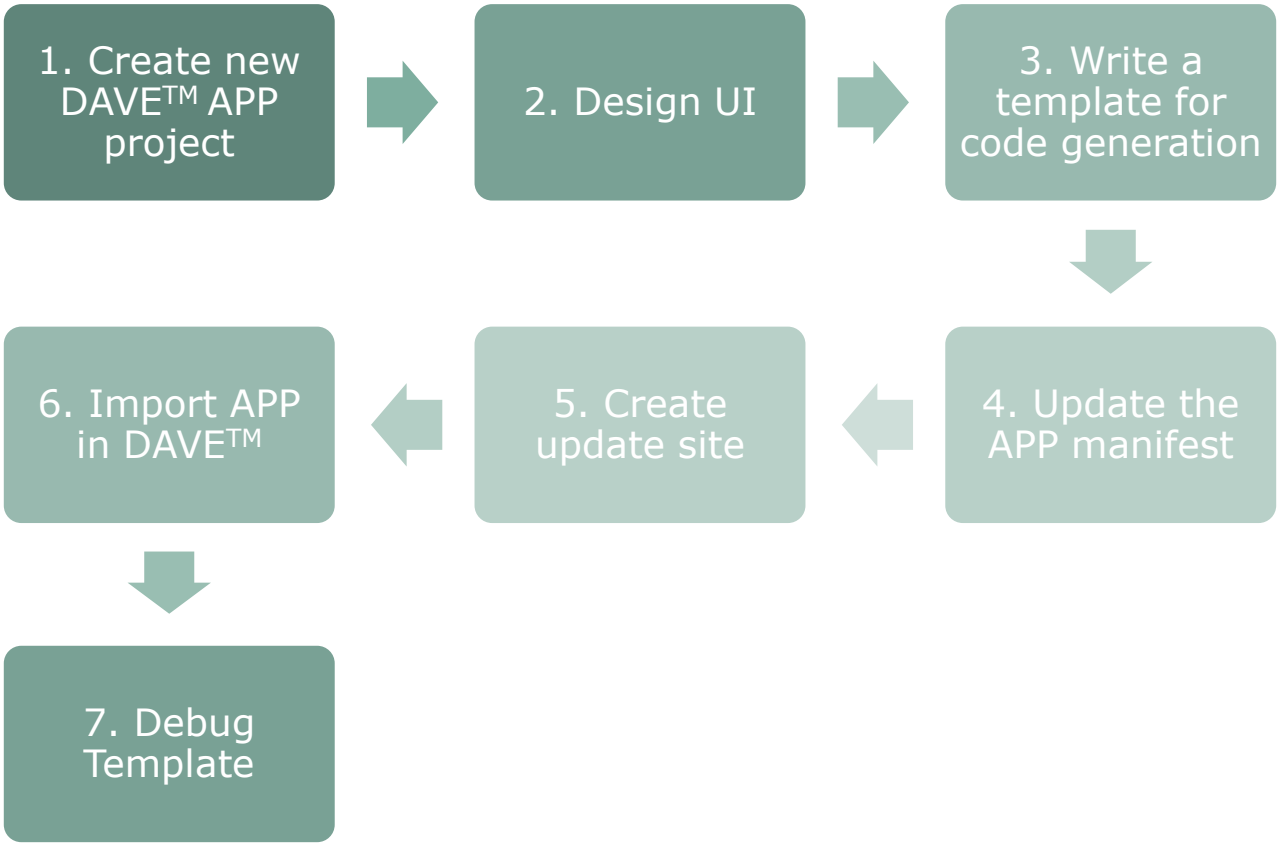
- Start DAVE™ SDK
- Enter path to workspace folder



DAVE™ SDK Workspace

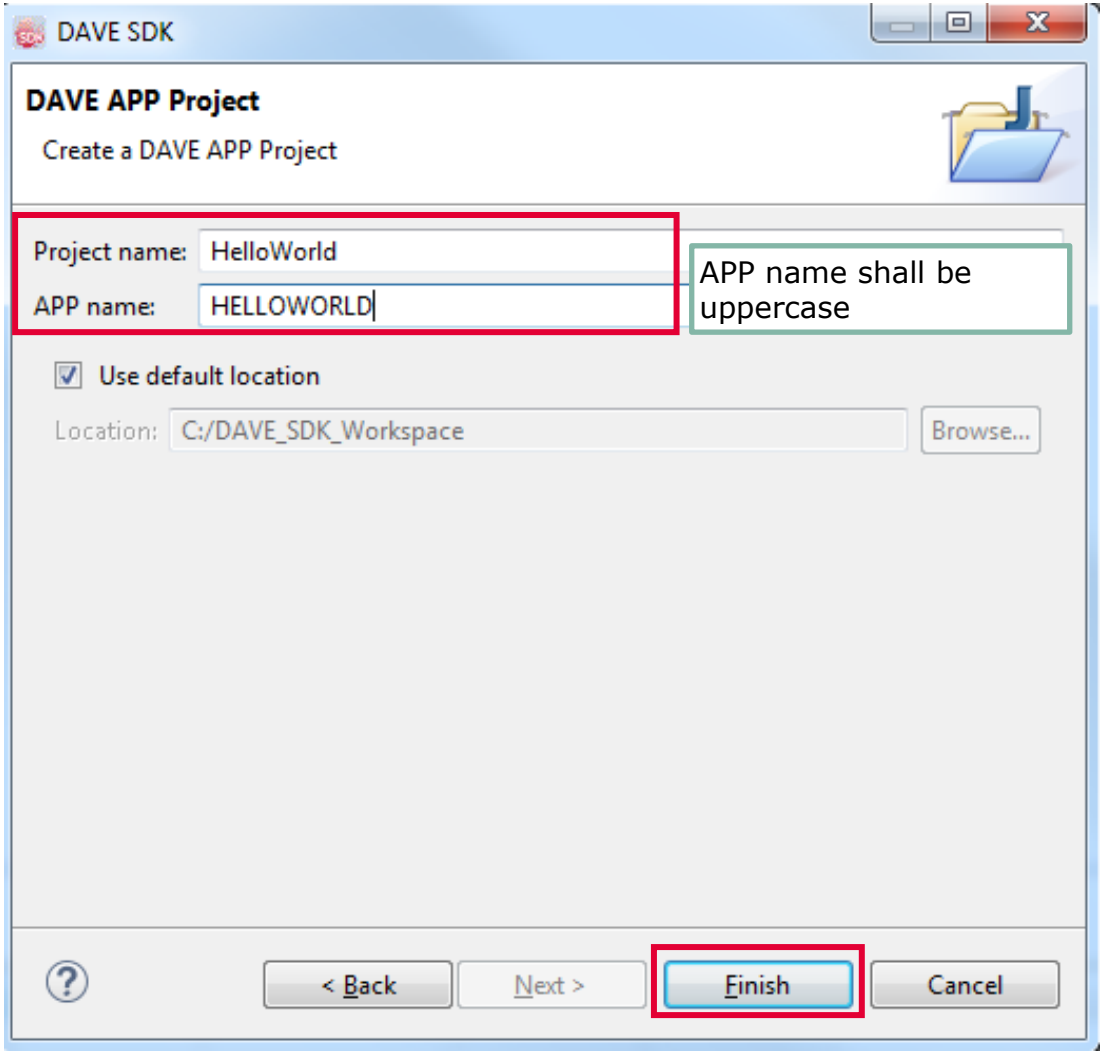


DAVE™ APP Development Flow

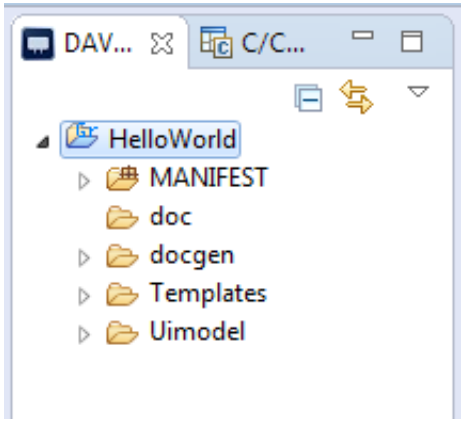


Create new DAVE™ APP Project

- Create DAVE™ APP Project
  1. Go to File → New → DAVE SDK Project
  2. Select APP Project
  3. Click Next
  4. Enter Project name and APP name



DAVE™ APP Project Folder

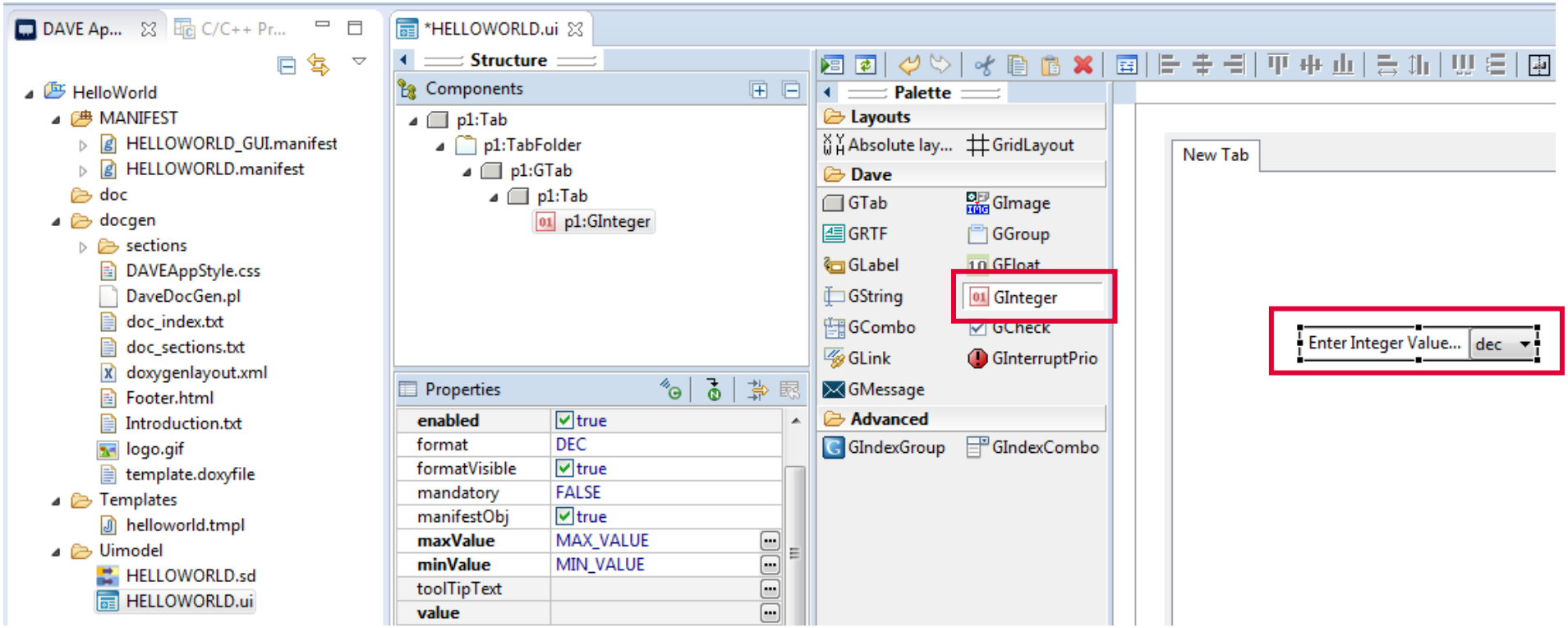


- MANIFEST
  - Groovy code used for User Interface logic
- doc
  - Generated APP documentation
- docgen
  - Source file required for APP documentation generation
- Templates
  - Code generation templates and files to be copied during code generation
- Uimodel
  - User interface configuration files

Design UI (1/4)

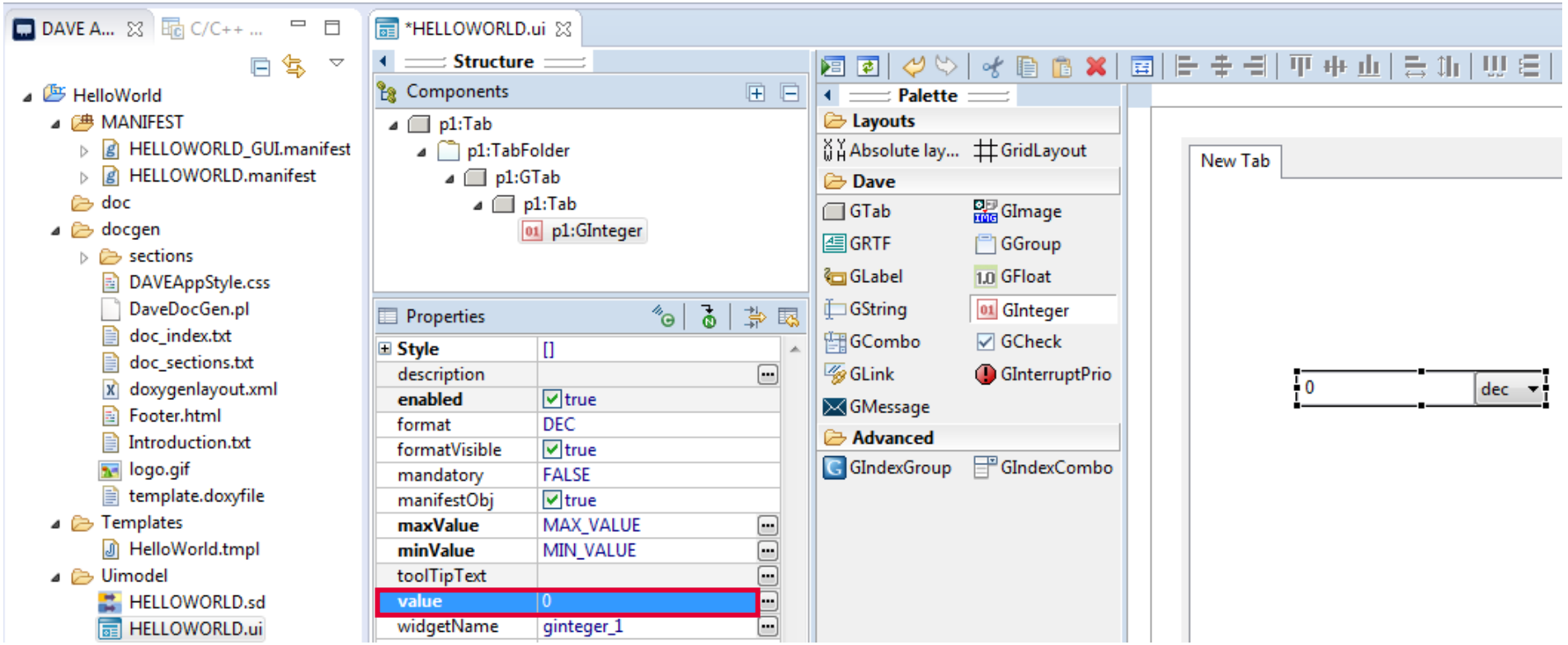
› Design DAVE™ APP user interface

1. Double-click HELLOWORLD.ui in Uimodel folder
2. Add a Ginteger widget



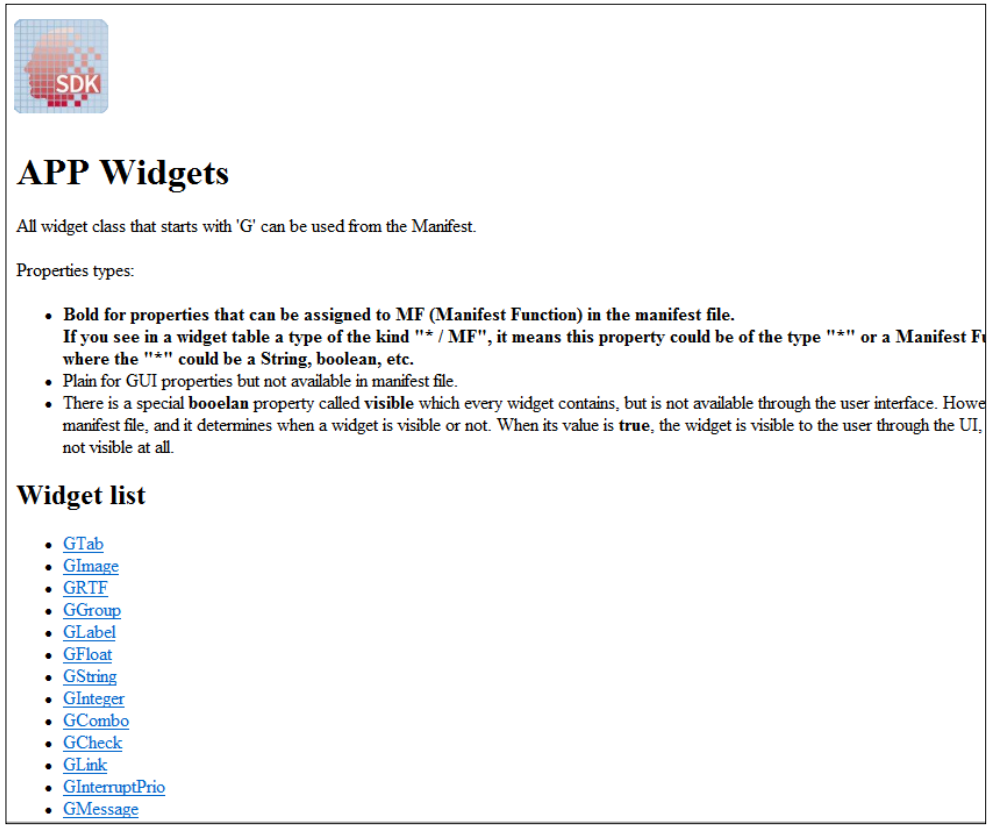
Design UI (2/4)

3. Set a default value for the widget



Design UI (3/4)

- APP widgets documentation
  1. Go to Help → Help Contents
  2. DAVE™ SDK User Manual → References → APP widgets



APP Widgets

All widget class that starts with 'G' can be used from the Manifest.

Properties types:

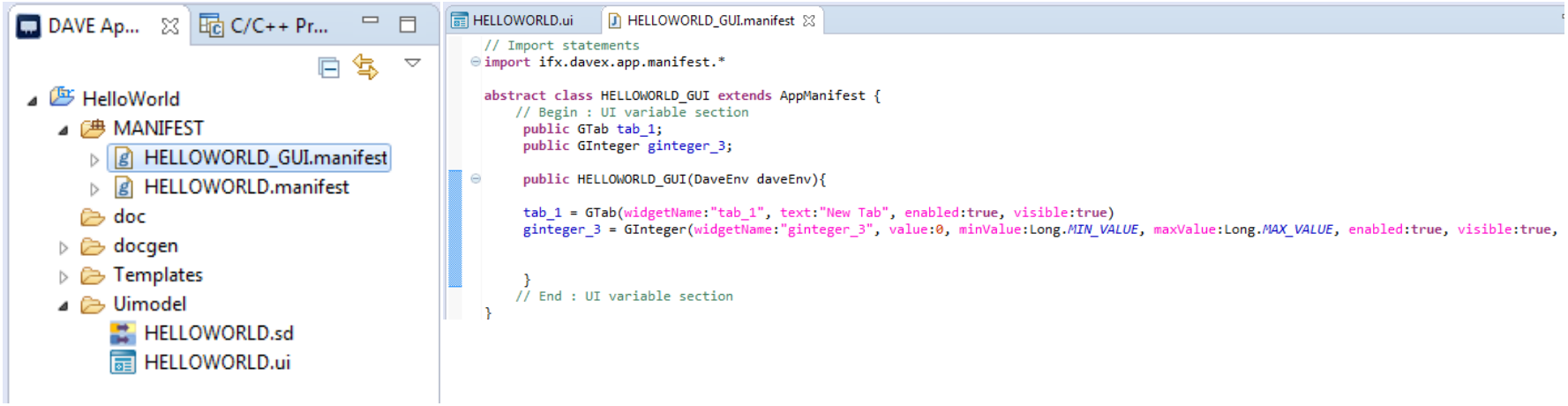
- **Bold for properties that can be assigned to MF (Manifest Function) in the manifest file.**
  If you see in a widget table a type of the kind "* / MF", it means this property could be of the type "*" or a Manifest Fu
  where the "*" could be a String, boolean, etc.
- Plain for GUI properties but not available in manifest file.
- There is a special **booelan** property called **visible** which every widget contains, but is not available through the user interface. Howe
  manifest file, and it determines when a widget is visible or not. When its value is **true**, the widget is visible to the user through the UI,
  not visible at all.

Widget list

- GTab
- GImage
- GRTF
- GGroup
- GLabel
- GFloat
- GString
- GInteger
- GCombo
- GCheck
- GLink
- GInterruptPrio
- GMessage

Design UI (4/4)

- Generate UI interface
  - Click  or CTRL+S to save the UI
  - At the same time, it will generate the HELLOWORLD_GUI.manifest
  - This file is the interface between UI and the user defined manifest HELLOWORLD.manifest



Write a template for code generation

- Remove existing template
  1. In Templates folder, right-click helloworld.tmpl
  2. Select Delete
- Add a new template
  1. Right-click Templates folder
  2. Select New → App Template
  3. Change file name to HelloWorld.tmpl
  4. Add the following content

```
for (HELLOWORLD app in appInstancesList) {
//Create a macro for each instances of the app
  out.print("""
      #define ${app.getInstanceLabel()}_MYINTEGER ${app.ginteger_1.value}"""
          )
}
```

  5. GUI values can be read directly from the UI bypassing the APP manifest

Update the APP manifest: softwareIDs

- Double-click HELLOWORLD.manifest in MANIFEST folder
- Define devices compatible with this APP by defining the softwareIDs APP property

```
/**
 * softwareIDs is used to declare which devices are compatible with
 * It is a map with key a string containing a pattern that shall ma
 * the device and as value the minimal version of the app.
 *
 *
 */

//Any step of XMC4500F144
//Any step of XMC4400F100
//Any step, any variant and any packages of XMC1300
def softwareIDs       = [
    "XMC4.5.00.F144.*":"1.0.0",
    "XMC4.4.00.F100.*":"1.0.0",
    "XMC1.3.*.*.*":"1.0.0"]

/**
 * Singleton property:
 * When it is true the app can be instantiated only once.
 */
def singleton     = false
```

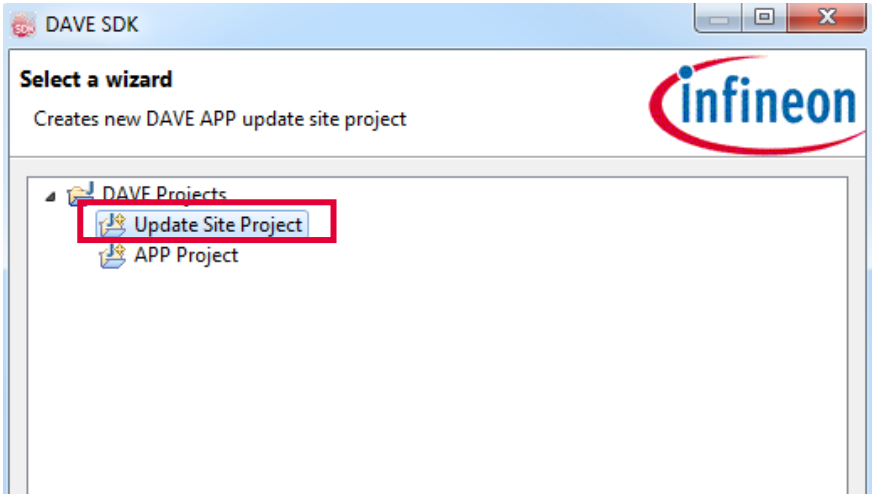
Update the APP manifest: generateFiles function

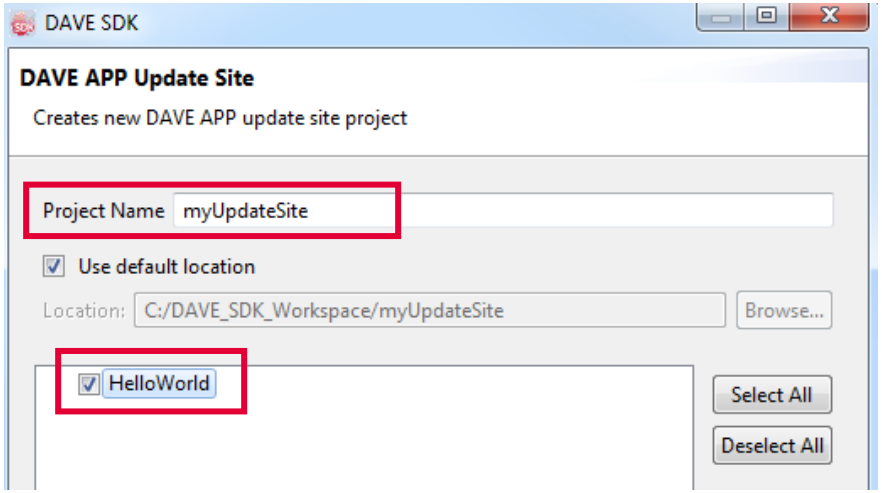
- generateFiles function is executed on code generation event
- User shall call a function to execute code generation based on templates

```
// App Constructor
public HELLOWORLD(DaveEnv daveEnv) {
    //TODO: Initialize manifest variables here
}

// File Generation
def generateFiles(){
generate("HelloWorld.tmpl","HELLOWORLD.h")
}

//
// Add main code here
//
```

Create update site (1/2)

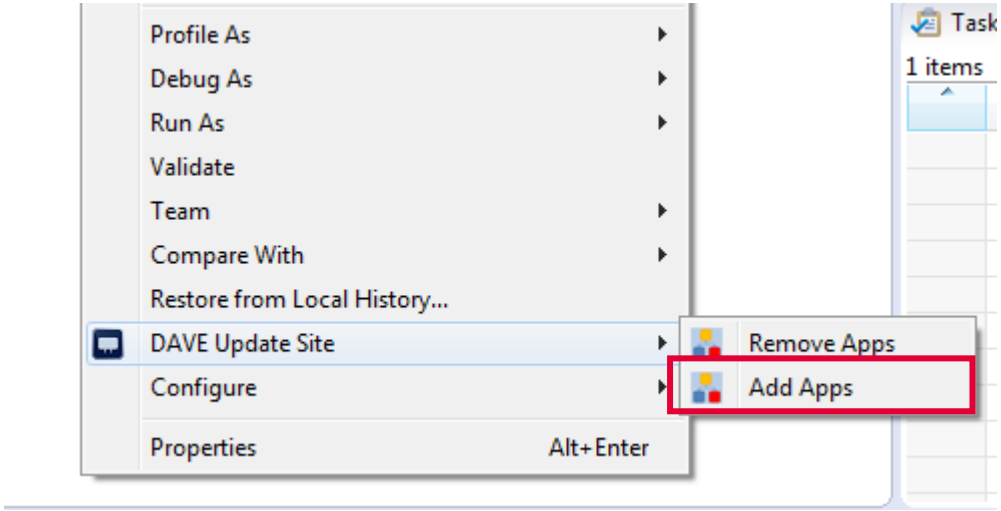
- Update sites are used to distribute a set of APPs and install them in DAVE™
- Create new update site project
  1. Go to File → New → DAVE™ SDK Project
  2. Select Update Site Project
  3. Enter update site Project Name
  4. Select HelloWorld APP





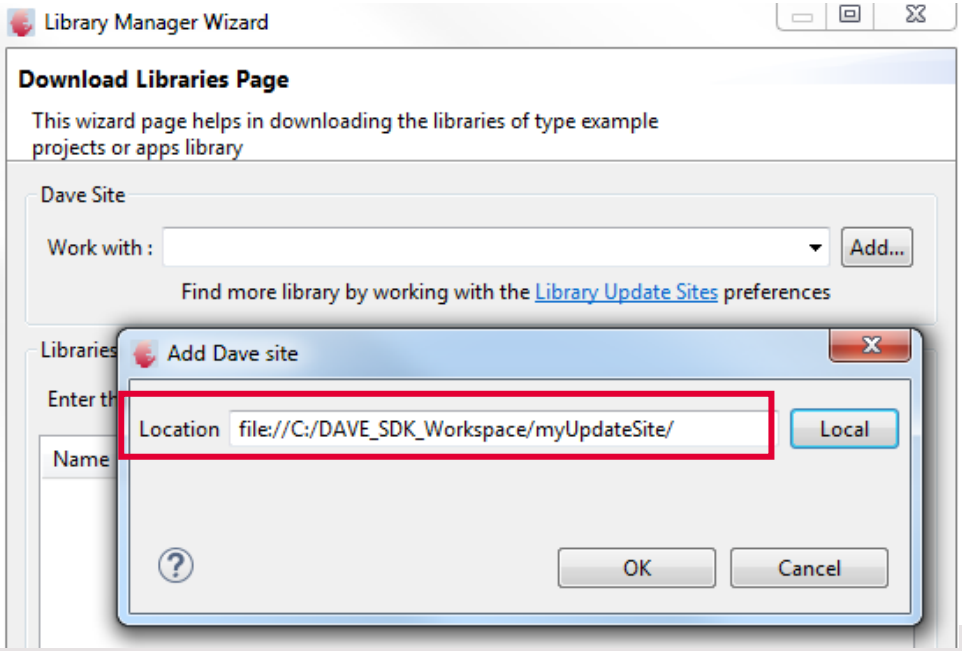
Create update site (2/2)

- Add APPs in workspace to update site project
  - Right-click on myUpdateSite folder
  - Select DAVE Update Site → Add Apps



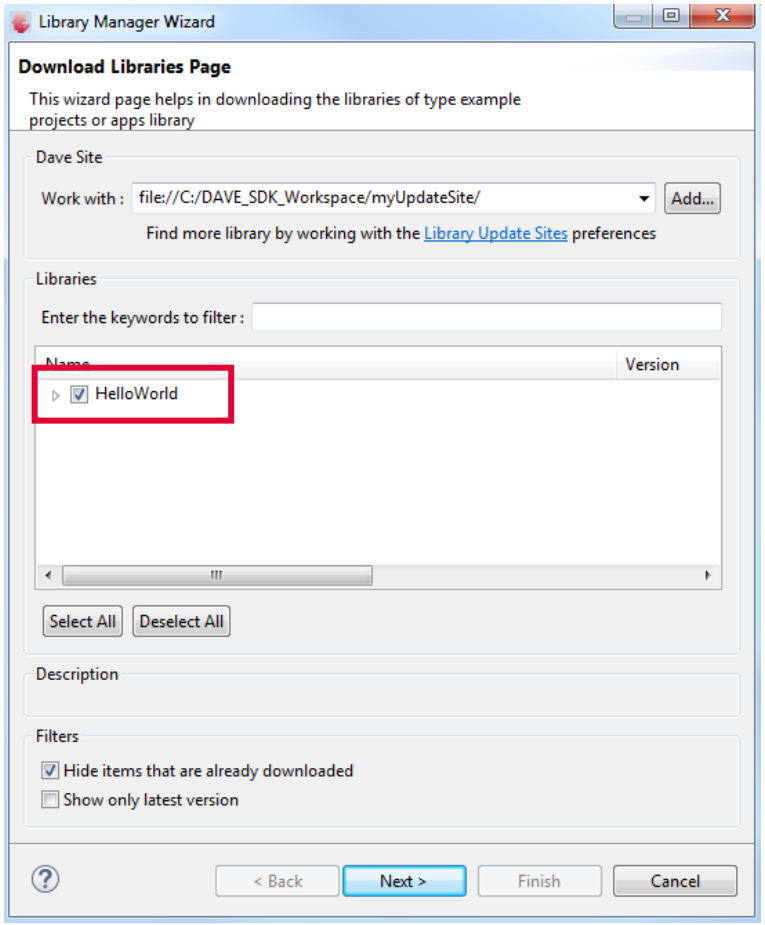
Import APP in DAVE™ (1/2)

- In DAVE™, it is possible to import APPs directly from DAVE™ SDK using the update site folder stored in the workspace
  1. In DAVE™ IDE, go to Help → Install DAVE APP/Example/Device Library
  2. Add Dave Site
  3. Click on Local and browse to update site folder



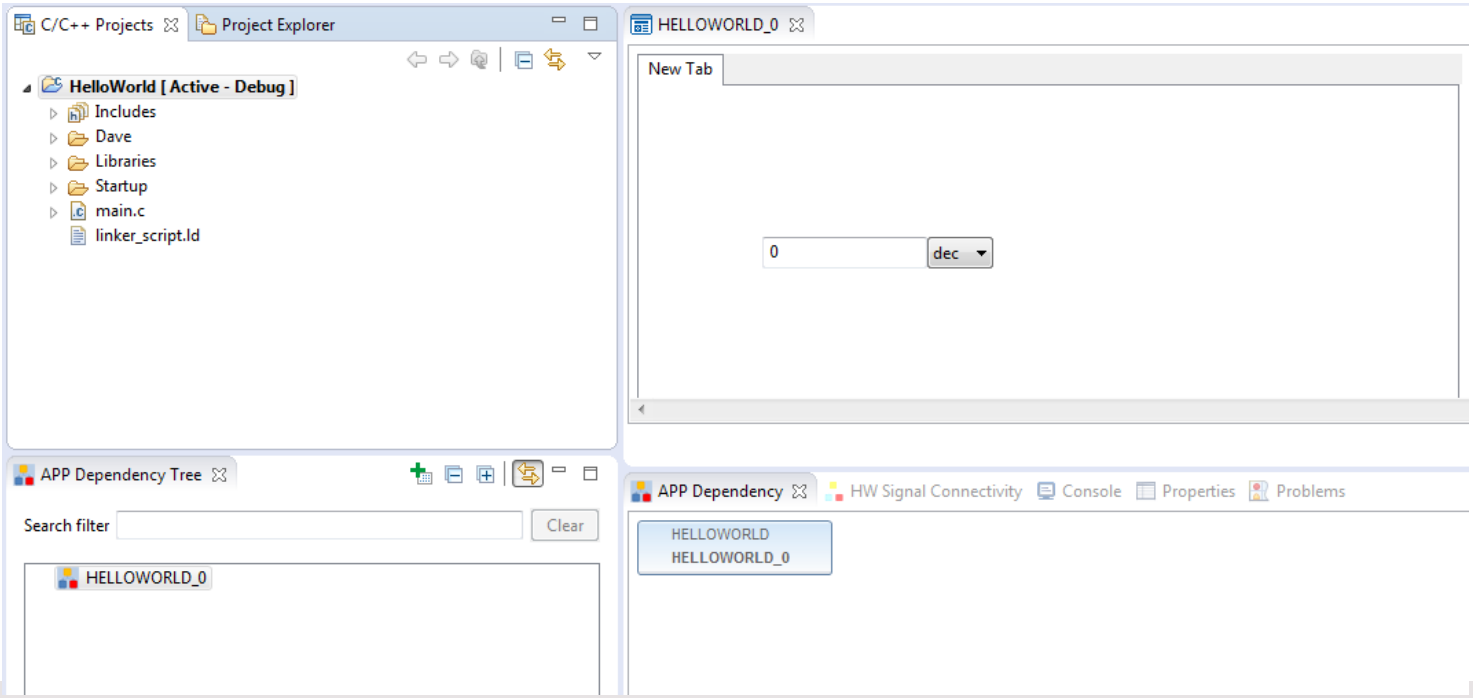
Import APP in DAVE™ (2/2)

4. Select HelloWorld APP → Click Next
5. Accept terms of licence agreement → Click Finish



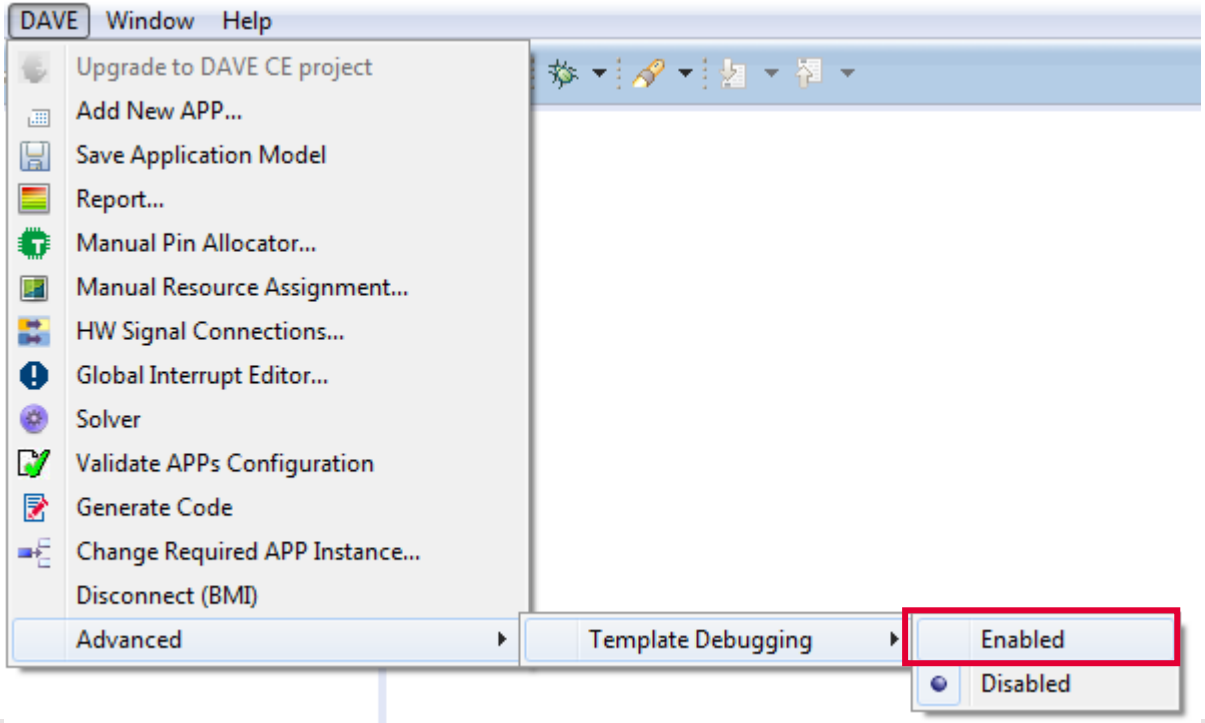
Debug Template (1/5)

- Before debugging code template, create DAVE™ CE project that uses HELLOWORLD APP
  1. Create a DAVE™ CE project
  2. Select target device that supports HELLOWORLD APP
  3. Add HELLOWORLD APP to project



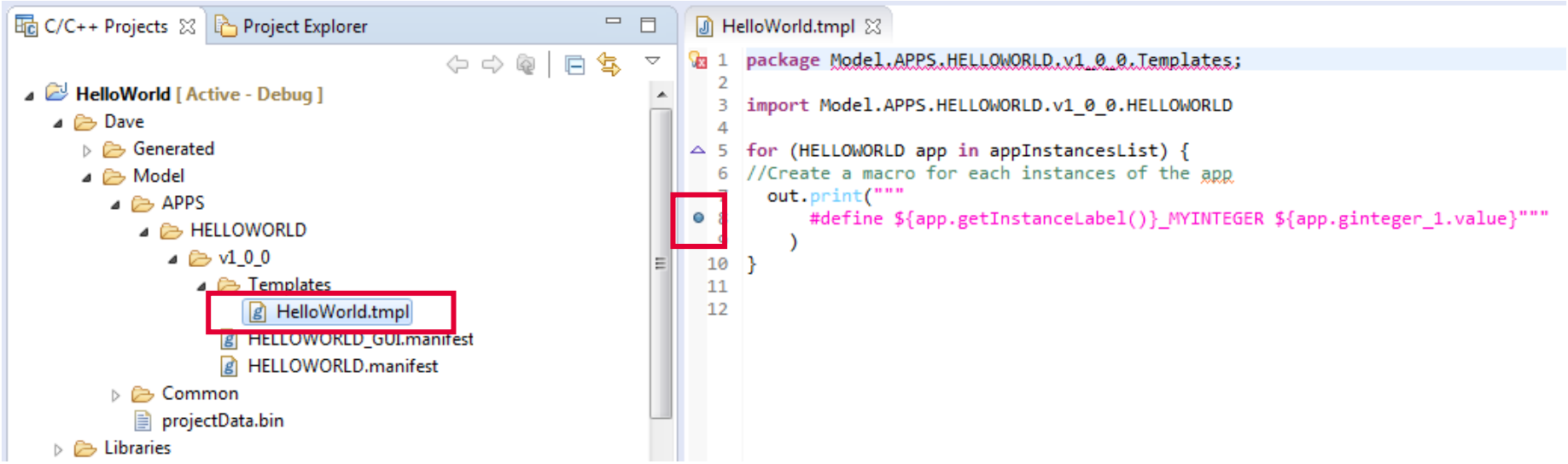
Debug Template (2/5)

- Enable template debug interface
  1. Go to DAVE → Advanced → Template Debugging
  2. Select Enabled
  3. This option makes visible the template debug profile



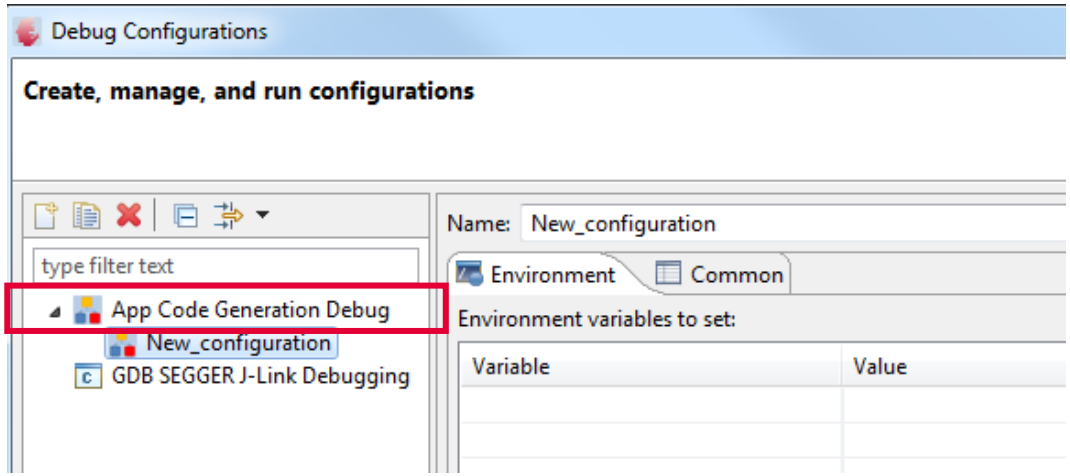
Debug Template (3/5)

- Set breakpoint in template
  1. Open HelloWorld.tmpl in Dave folder
  2. Double-click to add breakpoint at the line of code



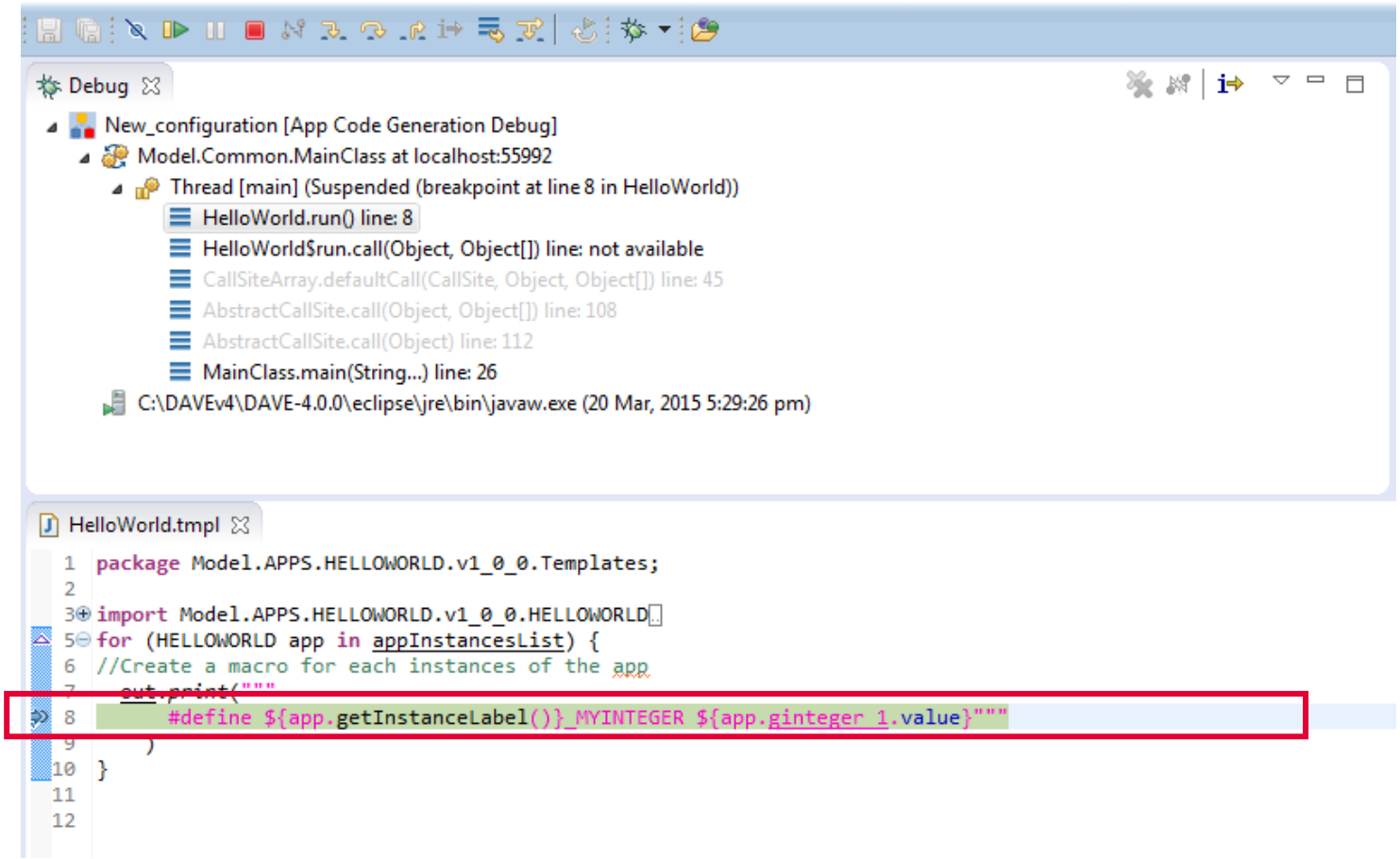
Debug Template (4/5)

- Start Debug Session
  1. Click  in the tool panel
  2. Setting the template debugging to enabled makes visible the App Code Generation Debug configuration
  3. Double-click App Code Generation Debug to create new debug configuration
  4. Click Debug



Debug Template (5/5)

5. DAVE will show Debug Perspective on the first breakpoint



Part of your life. Part of tomorrow.

Infineon