



Selected Topics in Java

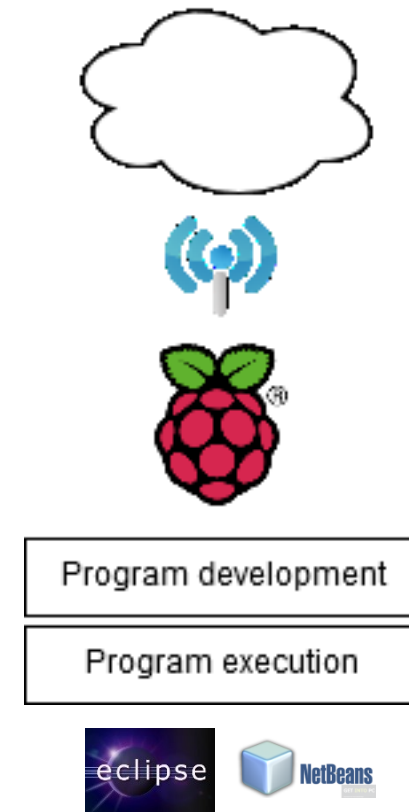
Java SE Embedded, Raspberry Pi

Martin Deinhofer, WS2018

Java SE Embedded, Recommended Development Setups

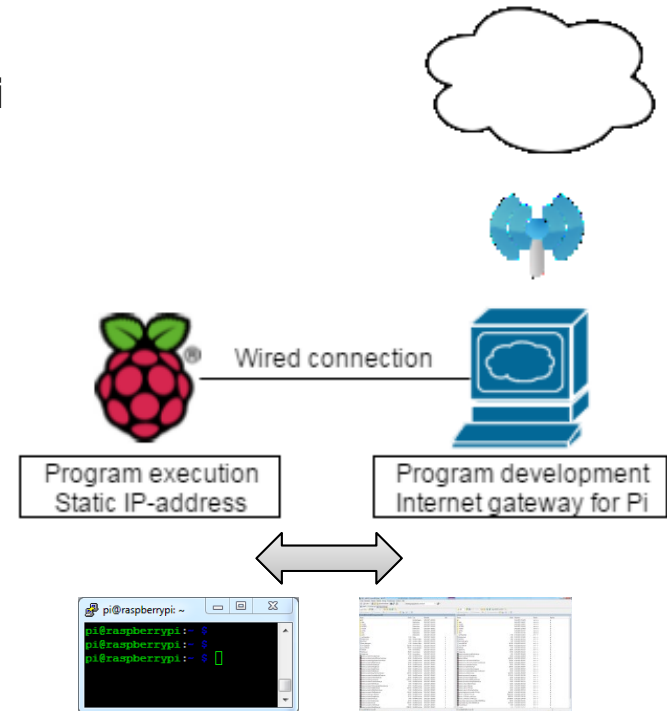
Development Setup – Only RPi

- **Program development and execution on RPi directly**
- Pros
 - Simple workflow (no file transfer,...)
 - Immediate testing
- Cons
 - Very slow (when using typical IDEs e.g. Eclipse)
 - → Use lightweight editors like geany and build scripts



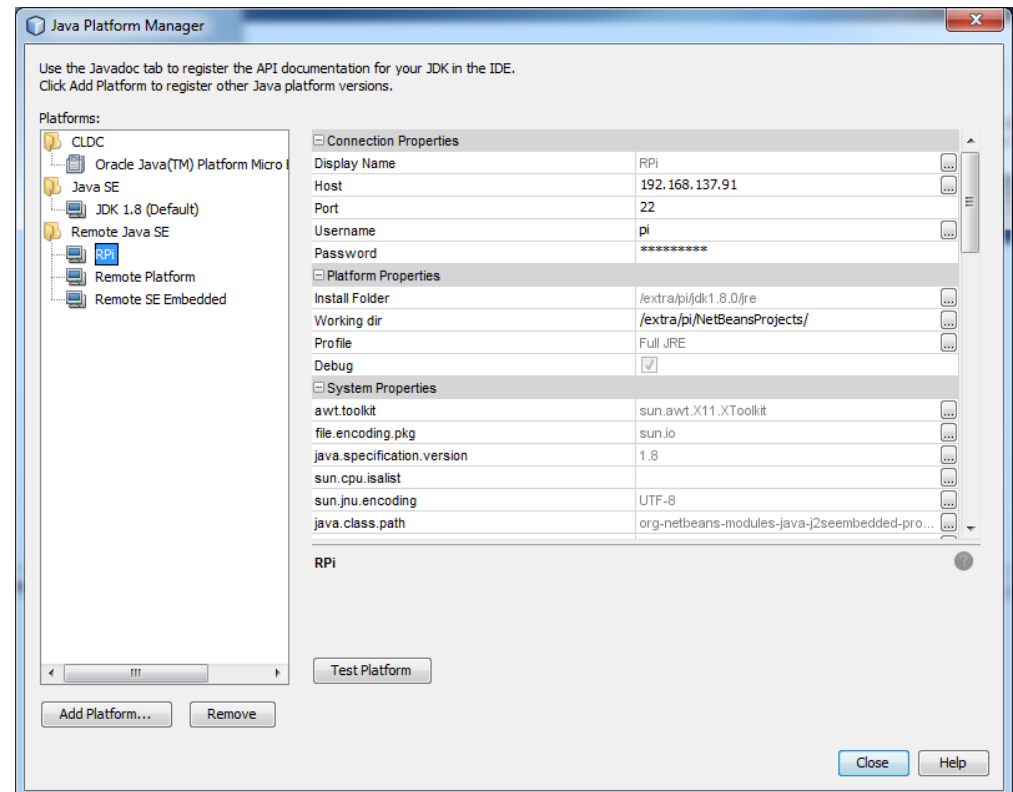
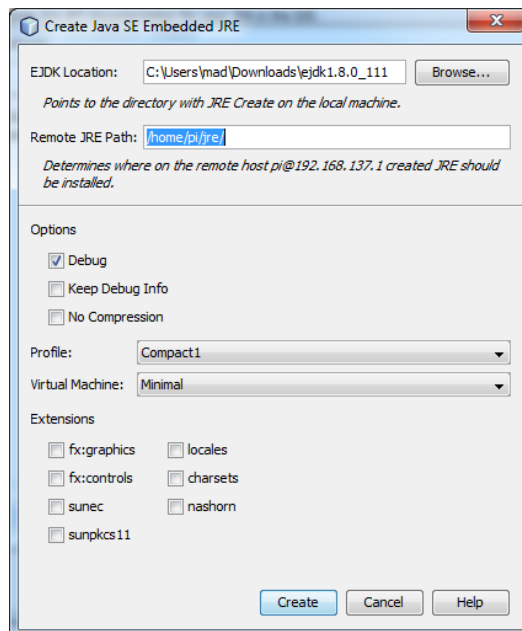
Development Setup – RPi+Development PC

- **Program development on development PC**
- **Program (compilation) and execution on RPi**
- Pros
 - Powerfeatures of IDE
 - fast
- Cons
 - Complex workflow (file transfer,...)
 - Several steps before testing
- Suggested Setup
 - Create project on Pi
 - Use remote file system share to edit files
 - Use remote shell to login to RPi
 - Use build script to compile and run program on RPi via remote shell



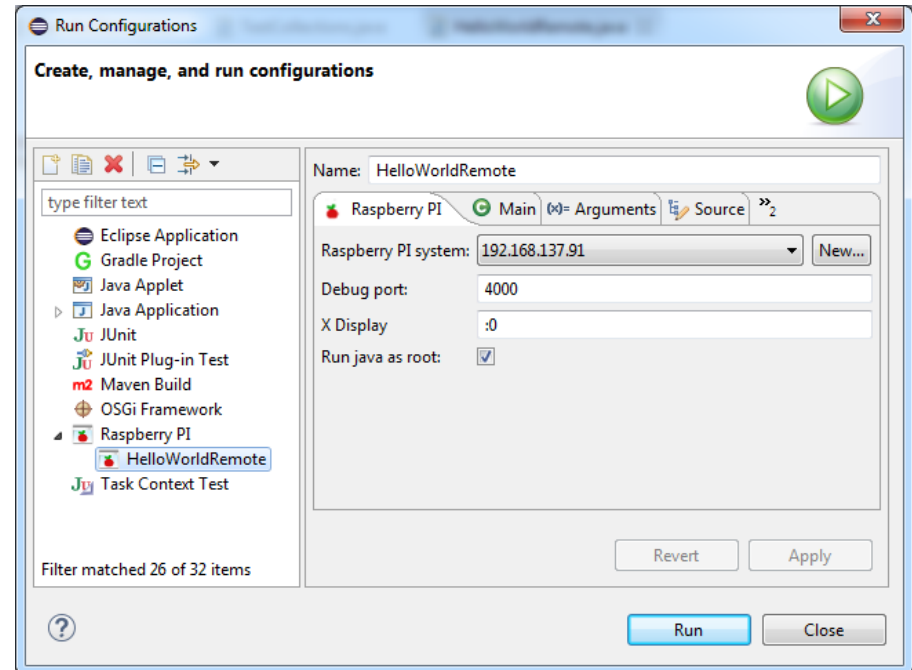
Workflow with NetBeans IDE

1. Create **Remote Java Runtime (SE or SE Embedded)**
2. Execute project on **Remote Java Runtime**



Workflow with Eclipse (LaunchPi)

- Plugin [LaunchPi](#): Uses scp, ssh to deploy and run program remotely



Workflow with Eclipse

(Remote System Explorer, SSH-Client)

- Edit files directly in built-in **Remote System Explorer** perspective
- Login to RPi with built-in SSH shell and execute build script.

Alternatives

- SSH-Client: [putty](#)
- File transfer: [WinSCP](#)
- Remote file system: [samba](#) or nfs mounted as network share

The screenshot shows the Eclipse IDE interface. On the left, the Remote System Explorer is open, showing a tree view for the IP address 192.168.137.91. Underneath, there are folders for 'Sftp Files', 'My Home', 'Root', and 'Ssh Shells'. The main editor area on the right displays a file named 'makefile' with the following content:

```

1 # Define common variables
2 CLASS_PATH = ../bin
3 JAVA_SRC_PATH = ../src
4 JAVA_HOME=/usr/lib/jvm/jdk-8-oracle
5
6 #Paths to libmcp23s17 and libpifac
7 LIB_MCP23S17=../../libmcp23s17
8 LIB_PIFACECAD=../../libpifacecad
9
10
11 # Define a virtual path for .class
12 vpath %.class $(CLASS_PATH)
13
14 all : libhello.so
15

```

Overview of Oracle Embedded technologies

- Oracle provides
 - Java SE and Java SE Embedded for ARM CPUs
 - **Java ME** for low foot print devices
 - **Java Card** for Smartcards

<p>Oracle Java SE Embedded</p> <ul style="list-style-type: none"> ▪ High performance virtual machine ▪ Enables development of reliable, portable embedded applications ▪ Deployment infrastructure ▪ Rich set of features and libraries 	<p>Oracle Java Embedded Suite</p> <ul style="list-style-type: none"> ▪ Brings middleware capabilities to embedded devices ▪ Persistent and reliable storage ▪ Web-based interfaces for device management 	<p>Oracle Event Processing for Oracle Java Embedded</p> <ul style="list-style-type: none"> ▪ Pushing fast data capabilities to edge devices ▪ Tailored for deployment on gateways ▪ Cascading deployments of event processing from device to data center
<p>Oracle Java ME Embedded</p> <ul style="list-style-type: none"> ▪ Optimized for small devices ▪ Support for on-the-fly application updates and remote operation ▪ Extend the lifetime, flexibility and value of embedded solutions 	<p>Java Card</p> <ul style="list-style-type: none"> ▪ Secure environment for applications on smart cards, and very small devices ▪ Deploy multiple applications on a single card or add new even after card is issued ▪ Unique tools for developing new products 	<p>Java ME</p> <ul style="list-style-type: none"> ▪ Robust, flexible environment for mobile and embedded applications ▪ Built-in network protocols and security ▪ Portable across many devices yet leverages each device's native capabilities

Java SE Embedded - Profiles

- Supports customization of Java VM
 - 3 profiles only containing a subset of Java API

Full SE API	Beans	Input Methods	IDL
	Preferences	Accessibility	Print Service
	RMI-IIOP	CORBA	Java 2D
	Sound	Swing	
	AWT	Drag and Drop	
	Image I/O	JAX-WS	
compact3	Security ¹	JMX	JNDI
	XML JAXP ²	Management	Instrumentation
compact2	JDBC	RMI	XML JAXP
compact1	Core (java.lang.*)	Security	Serialization
	Networking	Ref Objects	Regular Expressions
	Date and Time	Input/Output	Collections
	Logging	Concurrency	Reflection
	JAR	ZIP	Versioning
	Internationalization	JNI	Override Mechanism
	Extension Mechanism	Scripting	

Java SE Embedded – Command line

You can create a custom Java VM with Netbeans or via the `jrecreate.sh` command line tool

1. Download and extract [Java SE Embedded](#) to your pi
2. Execute the tool `jrecreate.sh`

```
pi@raspberrypi:~/ejdk1.8.0_111/bin $ JAVA_HOME=/usr/lib/jvm/jdk-8-oracle-arm32-vfp-hflt/ ./jrecreate.sh -d my-jre-compact1 -p compact1
Building JRE using Options {
  ejdk-home: /home/pi/ejdk1.8.0_111
  dest: /home/pi/ejdk1.8.0_111/bin/my-jre-compact1
  target: linux_armv6_vfp_hflt
  vm: minimal
  runtime: compact1 profile
  debug: false
  keep-debug-info: false
  no-compression: false
  dry-run: false
  verbose: false
  extension: []
}

Target JRE Size is 10,813 KB (on disk usage may be greater).
Embedded JRE created successfully
pi@raspberrypi:~/ejdk1.8.0_111/bin $
```

Breakout – Suggested Exercises

- Setup Raspberry Pi with PiFace Control and Display (PiFaceCAD) add-on board, according to this [tutorial](#).
- Use JNI to send the following two-line text to the add-on display from within Java:
**Java for Embedded
rocks!!**
- Use the C-libraries [libmcp23s17](#) and [libpifacecad](#) to interface the PiFaceCAD.

