



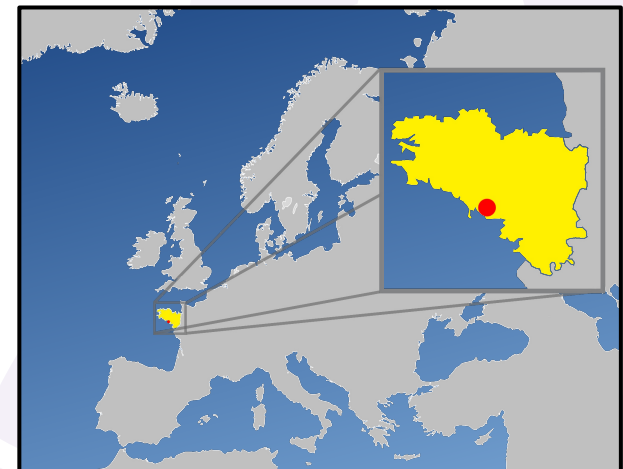
Leveraging Docker in Automotive projects based on AGL/GENIVI



Stéphane Desneux
CTO at IoT.bzh
<*sdx@iot.bzh*>

IoT.bzh

- Specialized on Embedded & IoT
- Contributing to AGL Project for Renesas
- Expertise domains:
 - System architecture
 - Security
 - Application Framework
 - Graphics & Multimedia
 - Middleware
 - Linux Kernel
- Located in Brittany, France



Agenda

- Light virtualization
- Containers for BSP builds
- Containers for Applications SDK
- Containers for CI & LTS
- Demo: AGL SDK for Renesas Porter board
- Limitations & Future enhancements
- Q&A

Light Virtualization [LV]



Light Virtualization








- Opposed to “**Full Virtualization**” which emulates a full machine (hardware + OS)
- A light virtual machine is also called a “**container**”: this is a kind of `chroot(2)` with some extra features
- A container runs its own processes based on its own binaries and libraries. But it relies on the Linux Kernel running on the host machine.
- Uses Linux **namespaces** to isolate the virtual system from the host system

see `unshare(2)`

LV: what's hype ?



Some software related to LV:

- Docker 
- Rocket (CoreOS) 
- Open Container Initiative 
- OpenVZ 
- LXC / LXD (Ubuntu) 

LV: historical usages



- Historically used for easy **deployment** of **Cloud services**
 - very fast startup (compared to full virtualization)
 - low overhead (less memory used, less storage)
 - better load balancing
 - optimized hardware resources usage
- Some **security** models also use containers to provide **isolation** for multiple resources (filesystem, network stack, processes, uids/gids ...)

Containers for BSP builds



Context



- Goal: build and maintain the **BSP** (Board Support Package) for a selected target platform/board
- BSP is based on **Yocto Project / Poky** and the main build tool is **bitbake**
- Integration team is responsible for **writing recipes** to build source packages and generating **binary packages** and **images**

Recipe for a good Base Container



- Take a **supported OS** for Yocto/Poky
AGL 1.0 based on Poky 1.7
→ use Debian 7.4 which is supported
- Add the **dependencies** for bitbake
→ gcc, binutils, python ...
- You get a Base Container which can be used to build **any Yocto based distribution**
- **Update** from time to time when required

Build'em all



- Take the **base container** (or a previous snapshot container)
- Add (or update) your **layers** according to current snapshot:
 - poky, openembedded, meta-foo ...
- Reuse the **caches** if available from a previous container
 - download cache
 - build cache (sstate-cache)
- **Build** a full image and/or SDK with bitbake:
 - move the results outside of the container
 - serve hot
- The new container is now the new "**snapshot container**"

Share the Snapshot Container



- **Commit and share** this container which includes:
 - Layers
 - Packages sources in download cache
 - Build cache (sstate-cache)
- Use for any kind of build that implies **bitbake**: binary packages, images, SDK ...
 - ready to use by any **platform developer**
 - useful in CI to **validate new patches**

Benefits



- **Stability:** A container allows to create simple, unbiased build environments which are officially supported independently of the host machine.
- **Isolation:** no more bugs caused by local conditions (specific host, OS, local package etc.) even if build recipes may still be buggy.
- **Uniformity:** the container used to build a snapshot can be shared across a community to allow anyone to rebuild binaries in the same conditions with deterministic results.
- **Performances:** optionally, the snapshot container may contain pre-built, shared caches to speed up most builds.
- **Time to market:** out-of-the box solution, which makes the integration easier

Containers for Applications SDK



Context



- **Application developer** creates native or HTML5 applications on top of the BSP for a specific board
- Yocto Project / Poky generates the base toolkit to **cross-compile** sources using the headers and libraries of a target image
- Developers often use an **integrated development environment** (IDE) to build, run and debug efficiently.

Building the SDK Container



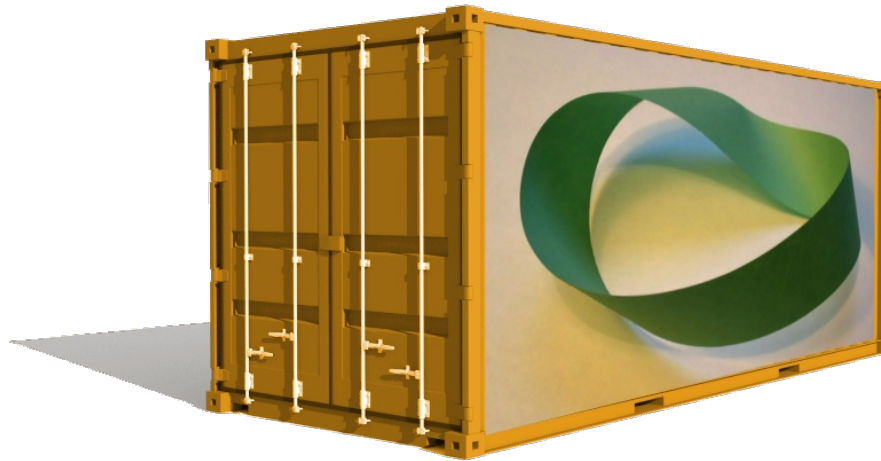
- Start with a good **distro** container
 - for AGL SDK, we took Debian 8.x
- Install the **BSP SDK** (cross-compiler, headers, libraries, sysroot) produced by the BSP Container
- Install **Eclipse IDE and plugins**, like the Yocto ADT plugin which allows cross-build and remote debugging on a target board
- **Commit** the new container: you get a SDK Container, which is ready to use by application developers.

Benefits



- **Synchronized with BSP:** the SDK container depends on the BSP and applications are built with the same tools as the ones used to build the BSP.
- **Uniformity:** all developers can share the same environment
- **Ubiquity:** the SDK container may run on the Cloud, on-premises, on developers hosts ...
- **Cost & Time to market:** applications can be developed faster and ready earlier.

Containers for Continuous Integration [CI] & Long Term Support [LTS]



CI: Deployment



- Easy **deployment** of predefined components:
 - Jenkins container
 - Gerrit container
 - ...
- **Failover** may be easier to manage
- **Replicate** CI infrastructure locally

CI: Validation builds



- Builds are required to **validate** new unmerged patches
- **BSP Containers** can be used to perform those builds easily:
 - Instantiate the container
 - Adjust layers revisions to include the patch to test
 - Run the build
 - Optional: automatically run sanity tests on the generated image
- **Accept or reject** patch (or simply report success and errors) based on results

CI: QA tasks



- **QA Plans** can be implemented with containers, to get more deterministic results
 - Results don't depend on the host platform running the tests
- Have a base QA container with essential QA tools
- Create **QA containers** based on QA plans
- Run the appropriate QA Container(s) on each new snapshot/target platform
- On AGL: the JTA QA framework uses Docker containers to run tests

Long Term Support



- Projects with long life span require the ability to **backport fixes** on old releases:
 - Automotive: **10 to 20 years support** to expect
Average cars age in Europe is 9.8 years
 - Even older for nuclear or military projects
- **Open Container Initiative** : an open format for containers will help to get the required life time for a container format and to get the ability to run it many years after its creation

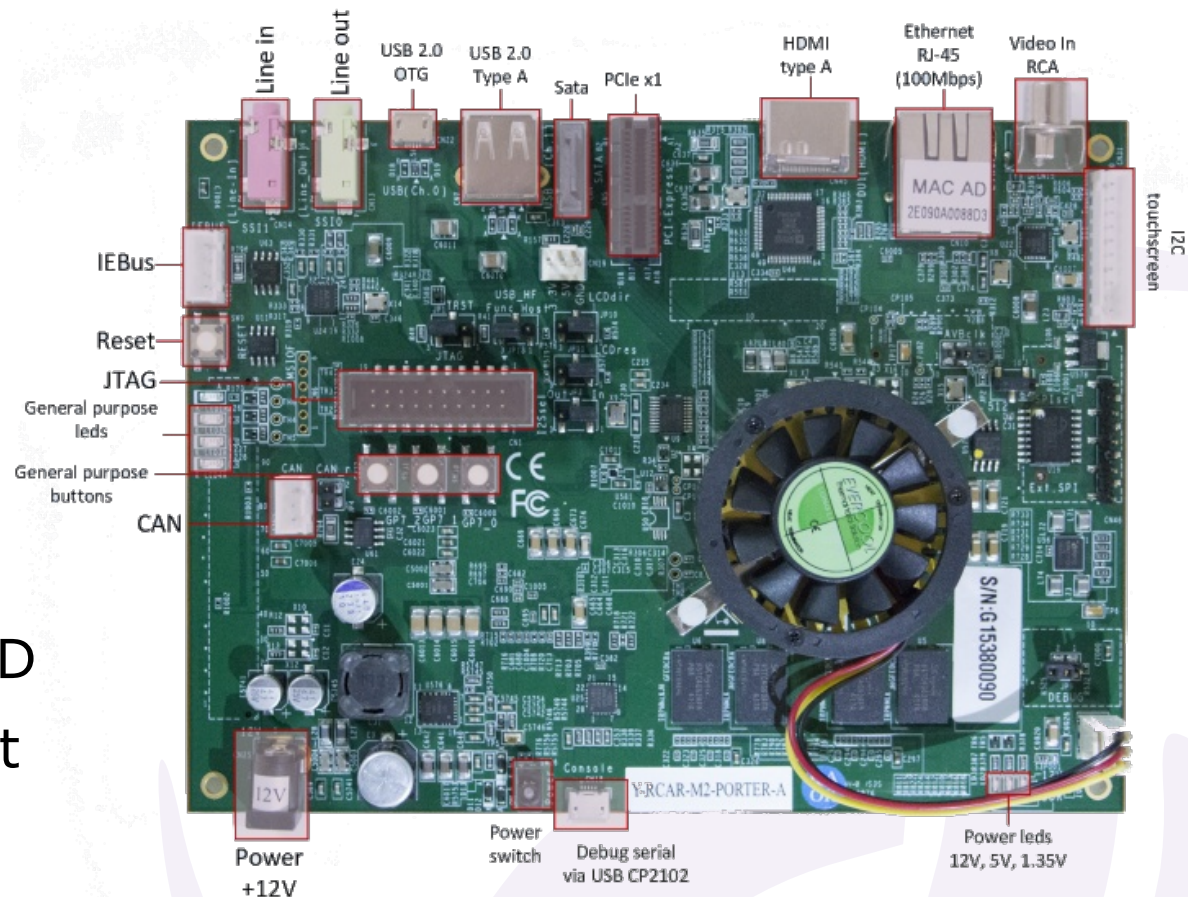
Demo with AGL SDK for Renesas Porter board



Renesas Porter Board



- R-Car M2 SoC
 - ARM Cortex-A15
 - Dual Core 1.5GHz
 - Multimedia Engine
 - GPU PowerVR SGX544MP2
- 2GB DDR3
- 2 Flash Mem Chips
- Ethernet
- Storage: SATA, SD, microSD
- Video: Analog In, HDMI Out
- Audio: In/Out
- USB 2.0
- CAN Transceiver



RENESAS

AGL SDK Initialization



- Load the SDK Docker image:

```
docker pull docker.iot.bzh/agl/snapshot-stable-sdk:1.0
```

- Instantiate a new SDK Container named '*aglsdk*':

```
docker run --publish=3389:3389 --detach=true \  
  --privileged \  
  --hostname=aglsdk --name=aglsdk \  
  -v /sys/fs/cgroup:/sys/fs/cgroup:ro \  
  docker.iot.bzh/agl/snapshot-stable-sdk:1.0
```

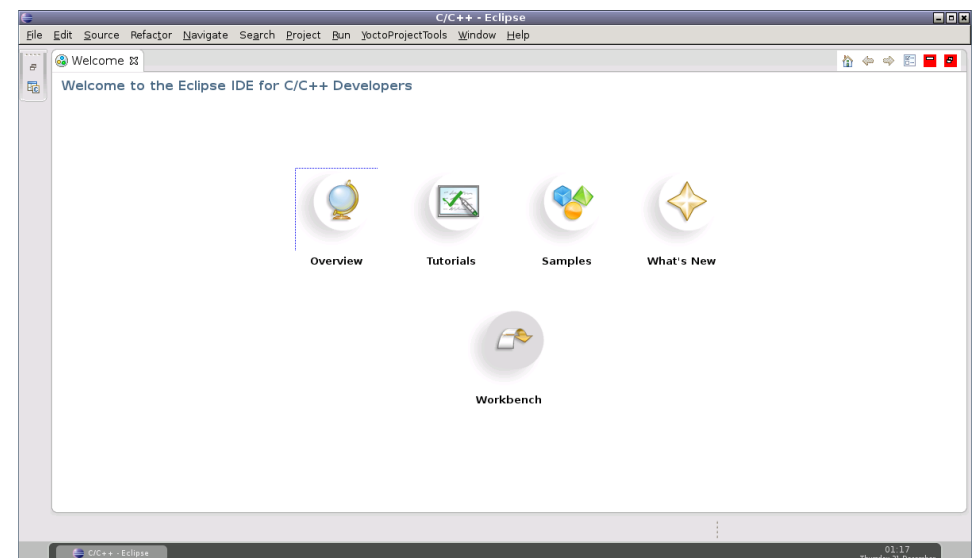
- Open a new RDP session on localhost:3389

```
xfreerdp -u devel -p devel -g 1200x700 localhost
```

Connect through RDP

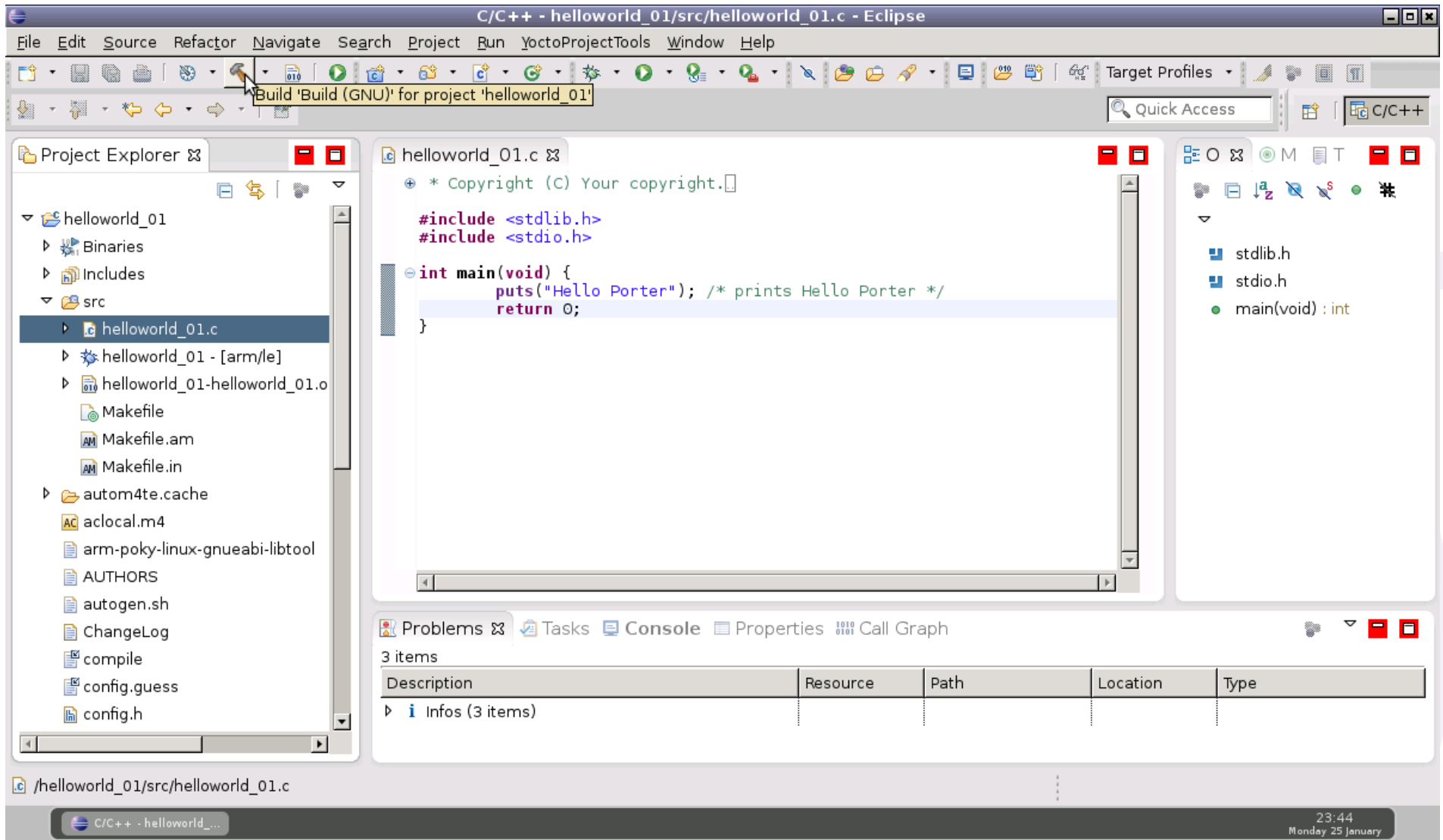


RDP session starting



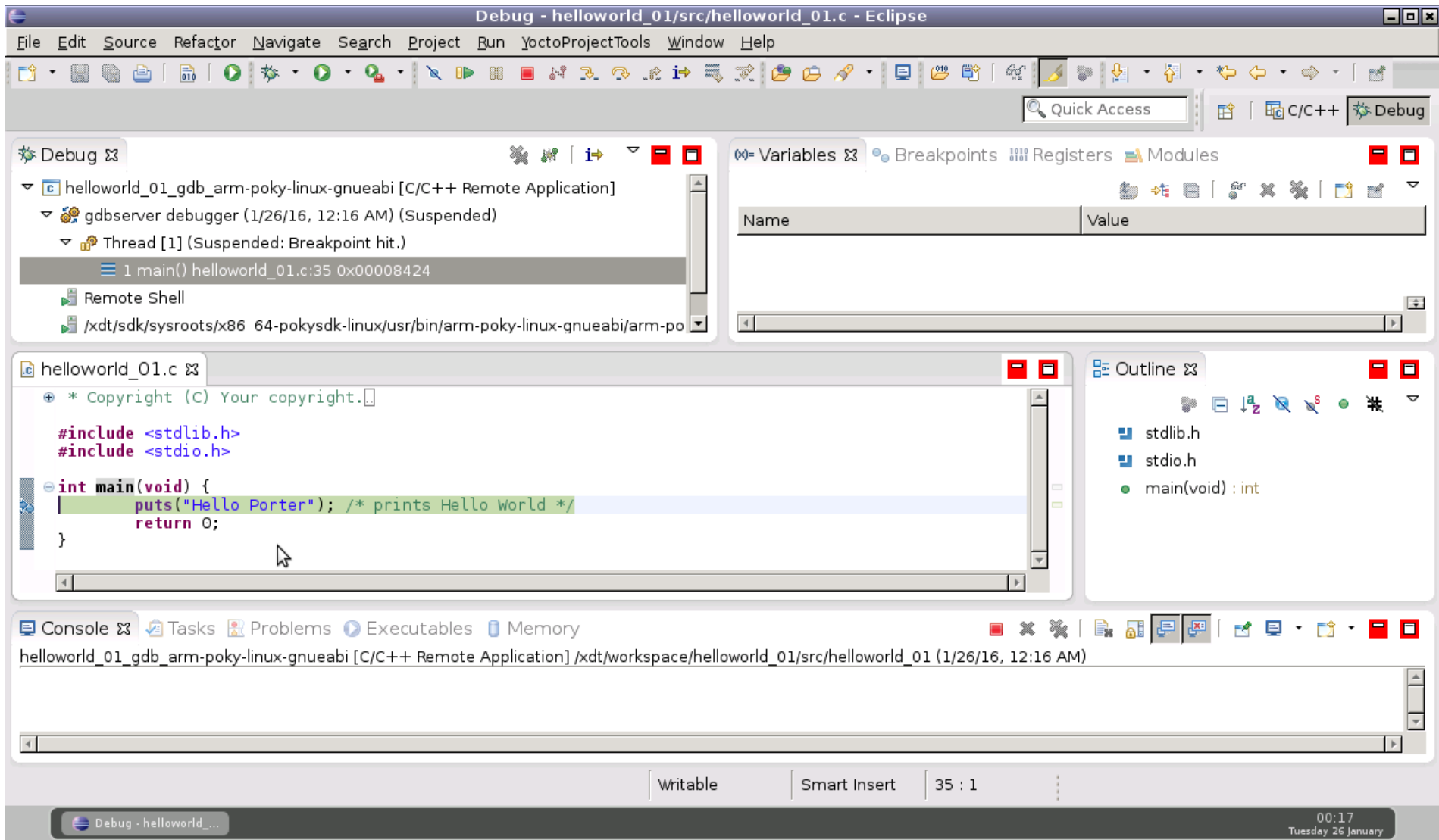
Eclipse IDE Initial Screen

Build a C/C++ program



Cross-compilation using Yocto & Eclipse

Remote debugging



Remote debugging on Eclipse using SSH and gdbserver

Limitations

Future enhancements



Limitations



- SDK Container
 - UID mapping in Docker is not supported yet
 - Persistent storage is not easy to setup and an external volume is not easy to share due to permissions conflicts
 - Eclipse IDE not in latest version
 - Target device access is more difficult for debug
- BSP Container
 - Docker and device mapping under Windows
 - loopback access not possible to flash SD cards

Future Enhancements



- BSP Container
 - Integrate features from Yocto 2.x
 - Optimize caches handling
 - Reduce storage size
- SDK Container
 - Switch to Web mode for IDE
 - Use next generation [Eclipse Che](#)
 - HTML5 Applications support
 - TCF Support for remote deploy & debug
 - Network boot of target boards

Q&A



Gulf of Morbihan, south of Brittany, France

Links - AGL

- Yocto Project: www.yoctoproject.org
- Automotive Linux: www.automotivelinux.org
- AGL 1.0 “Albacore” Release
<https://download.automotivelinux.org/AGL/release/albacore/1.0/>
- SDK Kickstart
<http://iot.bzh/download/public/2016/sdk/AGL-Application-SDK-Kickstart-on-Renesas-Porter-board.pdf>
- Eclipse IDE: eclipse.org

Links - Containerization

- Docker: docker.com
- LXC / LXD: linuxcontainers.org
- Open Container Initiative: opencontainers.org
- OpenVZ: openvz.org
- Rocket: coreos.com