
ORACLE®

Oracle Digital Assistant

The Complete Training

Custom Component Debugging

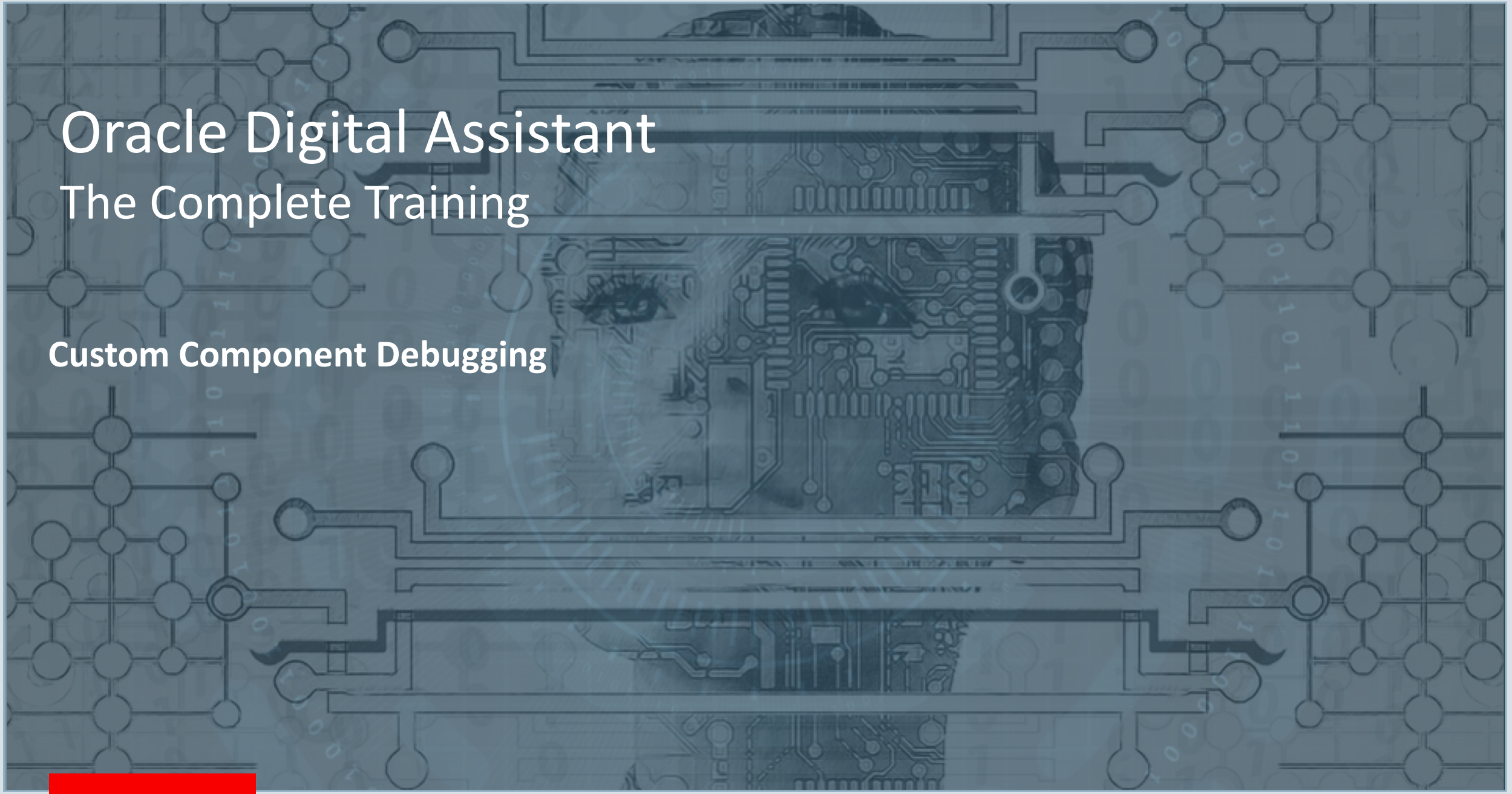


Image courtesy of pixabay.com

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Topic agenda

1. Debugging vs. logging
2. Debugging architecture
3. Debugging practices
4. Debugging with MS Visual Studio Code

Topic agenda

1. Debugging vs. logging
2. Debugging architecture
3. Debugging practices
4. Debugging with MS Visual Studio Code

Logging

- Logging is good to find problems that occur at runtime
  - Digital Assistant version 19.1.3 and later provide a diagnostic panel for displaying logs for locally deployed custom components
  - Mobile Hub and 3rd party Node containers have their own consoles for displaying logs
- Logging is not efficient enough for finding problems at design time
  - Not all problems are exceptions
    - Custom component doesn't render a response
    - Values returned by a custom component don't show in skill
    - Code logic is not getting executed
    - Broken dialog flow navigation after custom component was added

Debugging

- Monitors bot-component interactions
  - Access to bot message payload
  - Insights to variable states
  - Steps through code execution
- Use your JavaScript IDE of choice
  - E.g. MS Visual Studio Code, JetBrains Webstorm
- Run custom component service from your local computer
  - Connect from Oracle Digital Assistant skill
- If it works, then it continues working after deployment

Before you can debug custom component services

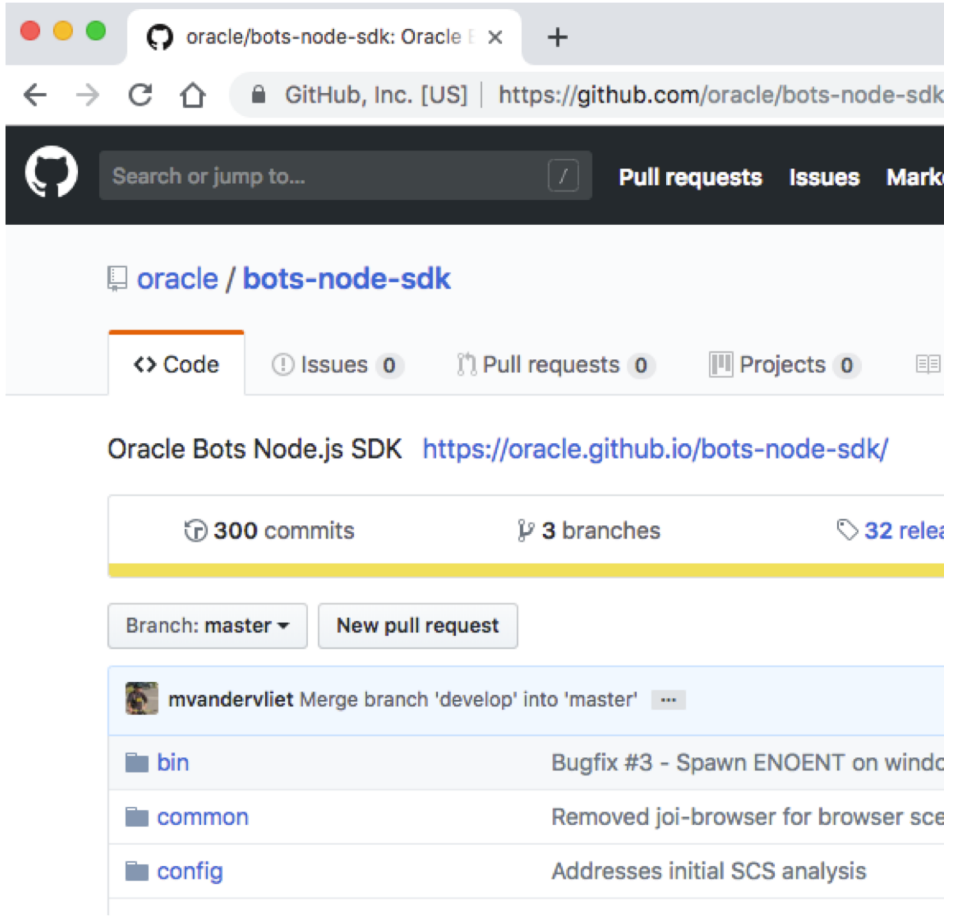
- Create local environment
  - Local runtime for custom component REST service
  - Install Oracle Bots Node.js
    - Command line to create custom component services
    - Includes local runtime
- Make local machine accessible from the Internet
  - Oracle Digital Assistant needs to access local component service runtime URL
- Have JavaScript IDE installed that supports Node debugging

Topic agenda

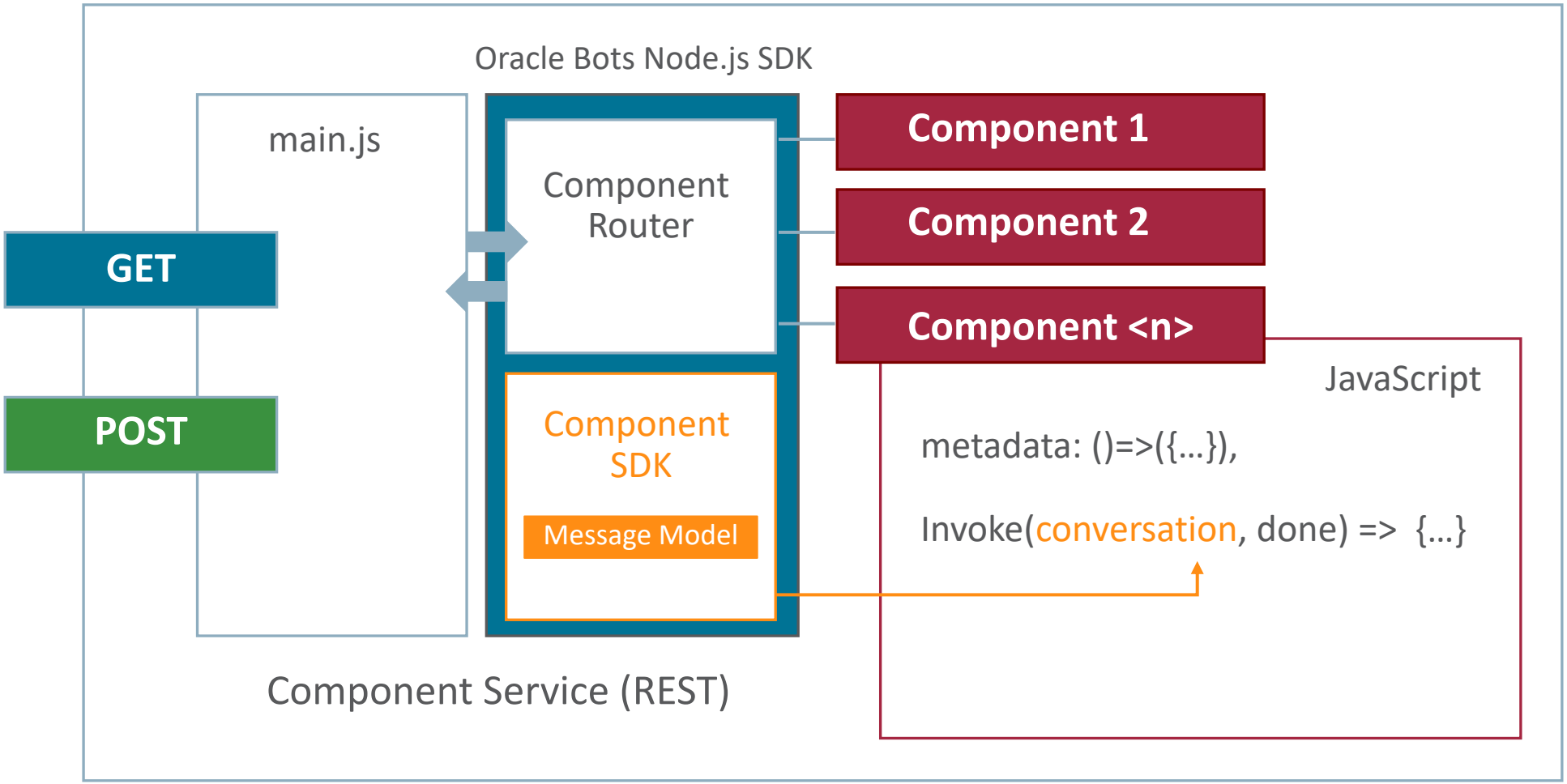
1. Debugging vs. logging
2. Debugging architecture
3. Debugging practices
4. Debugging with MS Visual Studio Code

About Oracle Bots Node.js SDK

- Provides local environment for
  - Component development
  - Component debugging
  - Packaging for deployment
- NPM installable
- Command line to
  - Create custom component service project
  - Create custom components
- Adds bots custom component SDK



Oracle Bots Node.js SDK component service architecture



For debugging, you **local machine must be accessible from** Oracle Digital Assistant running on **the Internet**

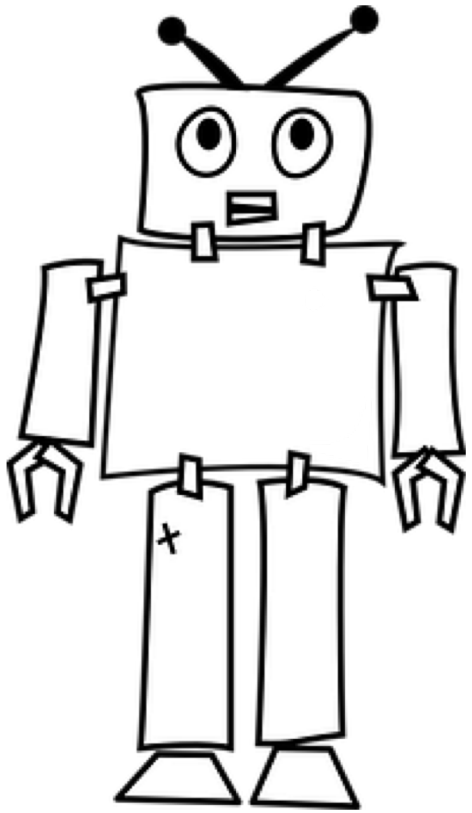
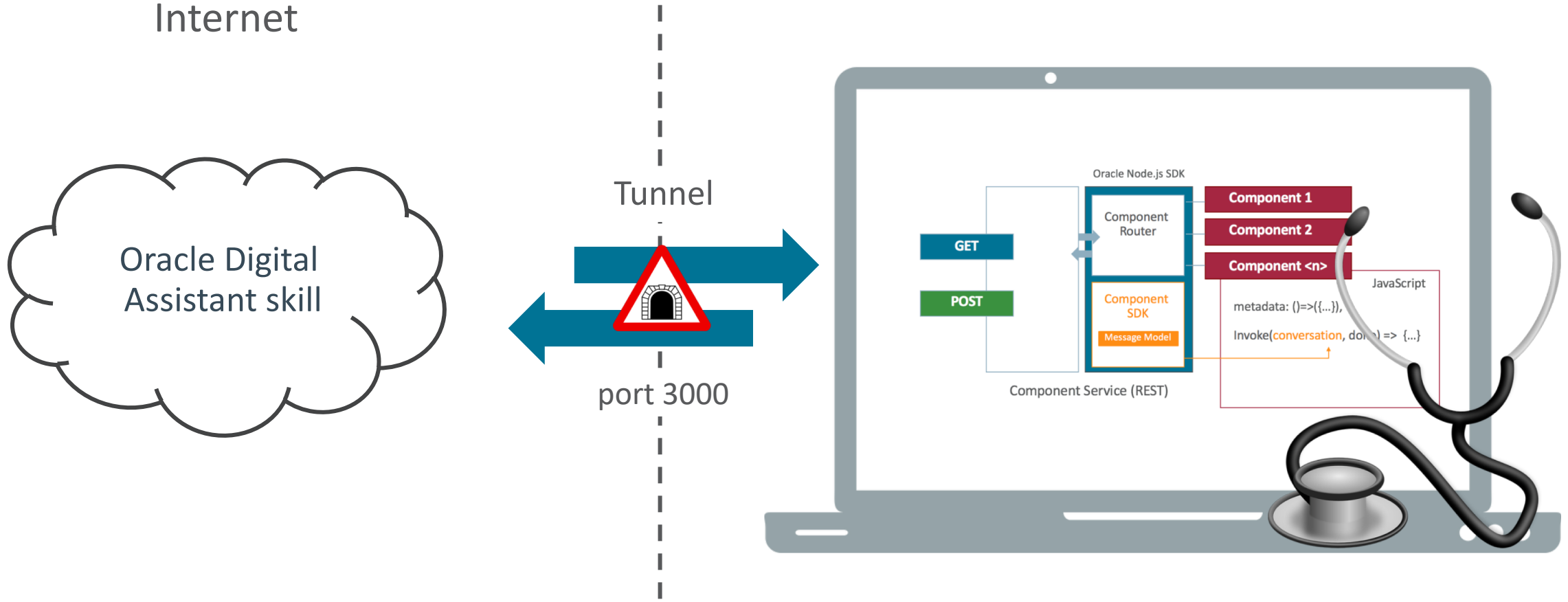


Image courtesy of pixabay.com

Debugging Architecture



Popular tunneling options

- ngrok
  - Well known
  - Doesn't require account
  - https://ngrok.com/download
- localtunnel
  - GitHub
  - Doesn't require account
  - https://github.com/localtunnel/localtunnel

Hints and tips

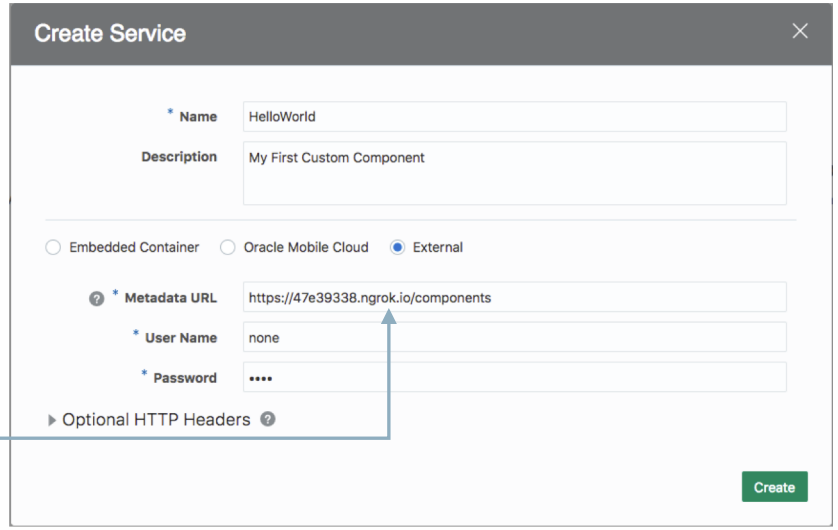
- Generally speaking, ngrok works on Windows and Mac
  - Oracle Windows images (OBI) don't allow the use of ngrok
  - When ngrok doesn't work, use "localtunnel" instead instead
- Ensure you have direct Internet access
  - Public or home network
  - Mobile hotspot
- If you are behind a proxy
  - Get the external IP address of your proxy (http://www.whatismyproxy.com/).
  - In the terminal window where you'll be using ngrok enter
    - export https_proxy=http://<external ip>:80
    - export http_proxy=http://<external ip>:80

Topic agenda

1. Debugging vs. logging
2. Debugging architecture
3. Debugging practices
4. Debugging with MS Visual Studio Code

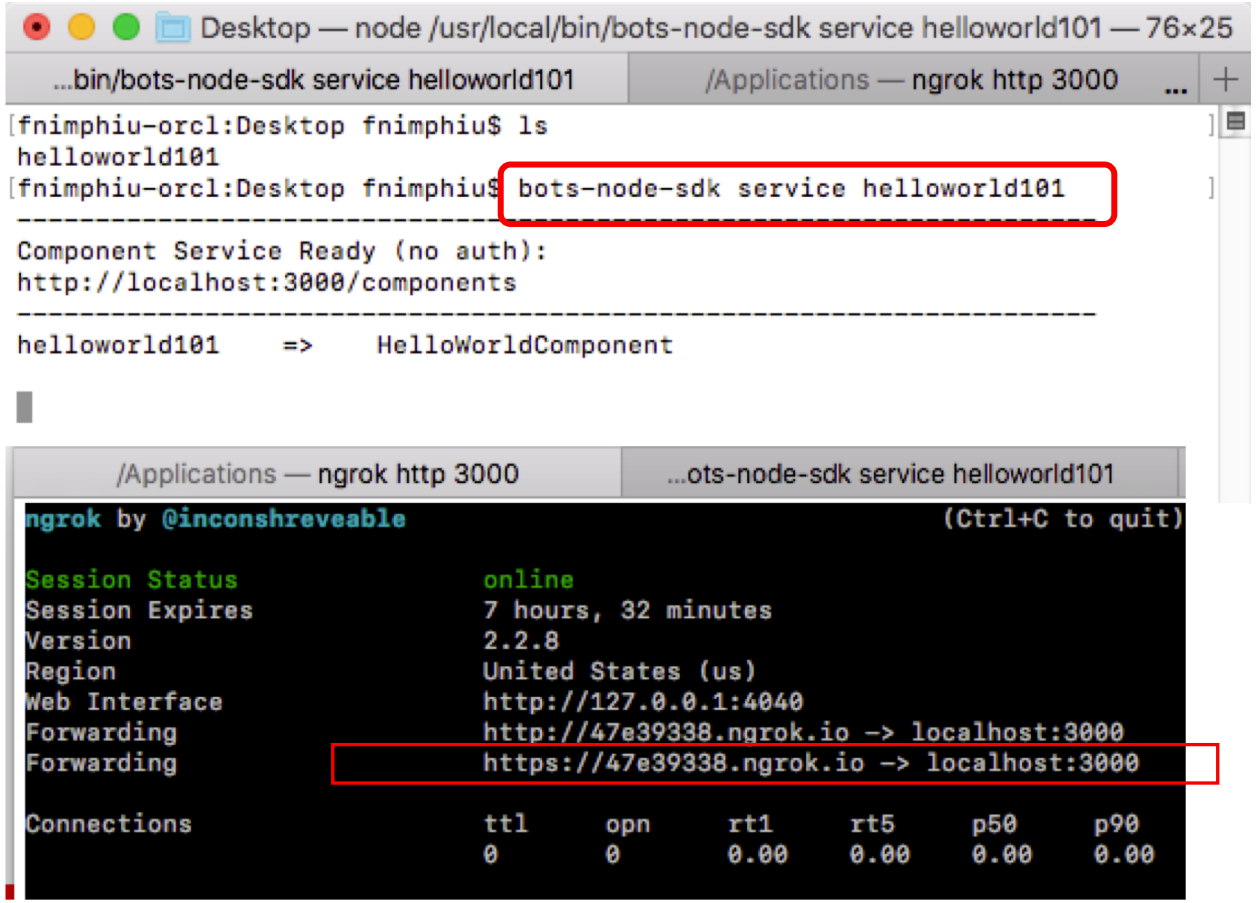
How-to debug custom component services

- Run Node server in Oracle Bots Node.js SDK
- Configure tunnel to expose port 3000
- Register component service 'External' service
  - Local runtime is an 'External' deployment
  - https://tunnel-url/components
- Connect JavaScript IDE debugger to local node server process
- Set breakpoints and use conversation tester in skill to start the debug session



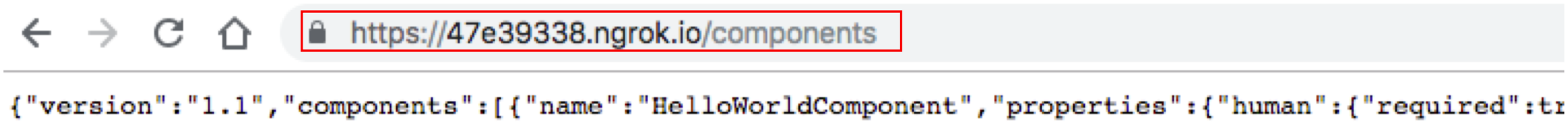
Starting the Node server on the local machine

- On local machine, start component service from parent folder
  - E.g. bots-node-sdk service helloworld101
- Open tunnel for port 3000
  - E.g. ngrok

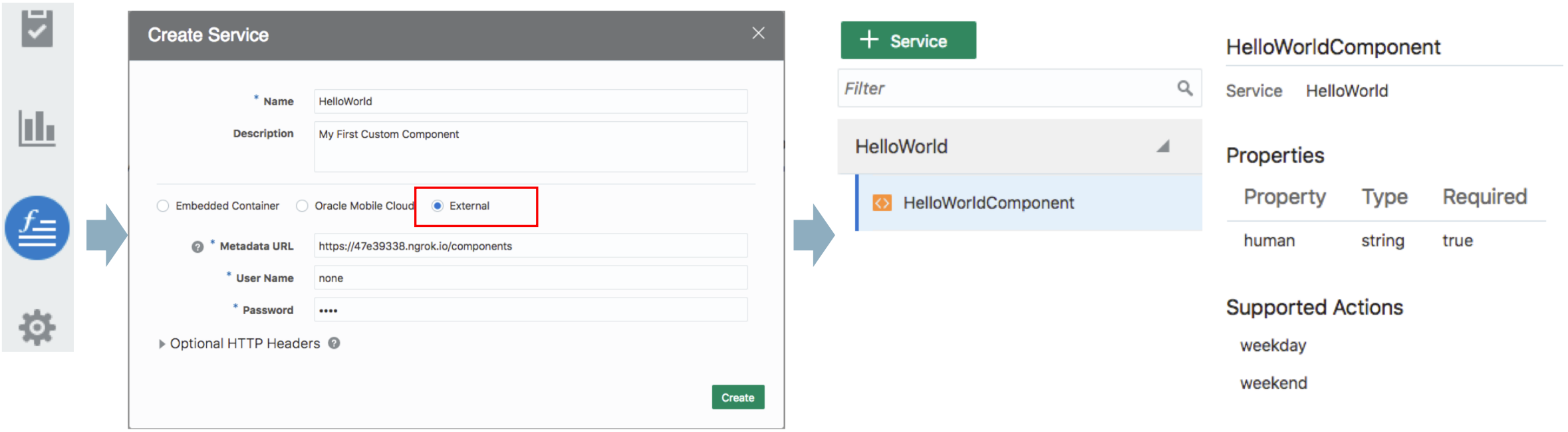


Testing the custom component service access in browser

- Type https://<tunnel URL>/components
  - Displays list of components hosted by component service
- Be aware
  - Tunnel URL changes with each re-start of the tunnel



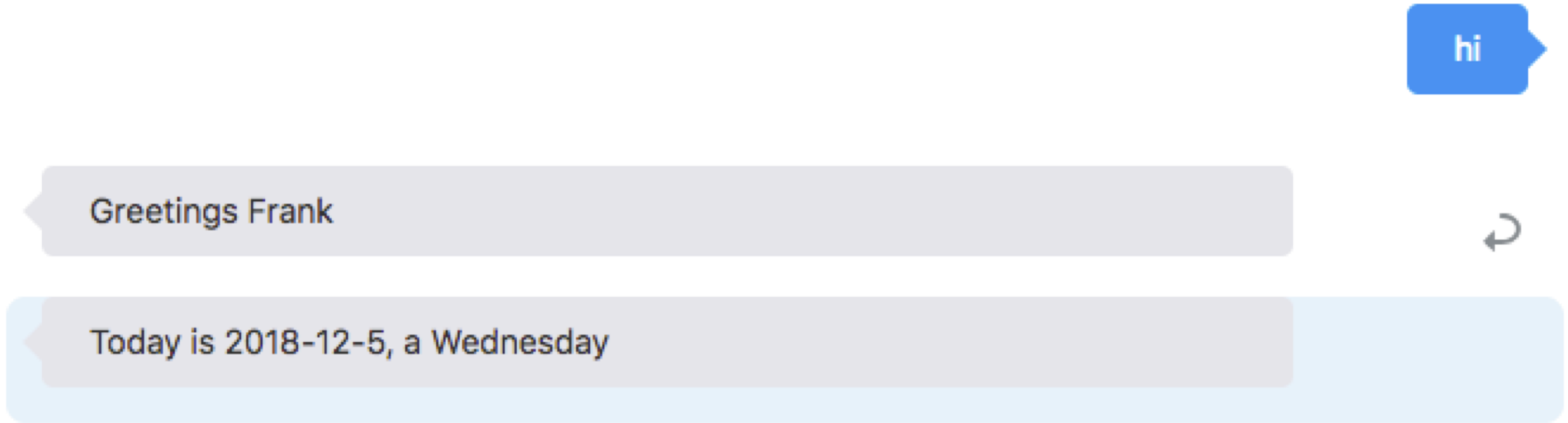
Register custom component service in skill



- Configure custom component service using ngrok or local tunnel URL
  - https://47e39338.ngrok.io/components
- No authentication required (just put something into the username and password field)

Dialog Flow configuration & runtime output

```
states:
  askGreeting:
    component: "HelloWorldComponent"
    properties:
      human: "Frank"
    transitions:
      return: "done"
```



Topic agenda

1. Debugging vs. logging
2. Debugging architecture
3. Debugging practices
4. **Debugging with MS Visual Studio Code**

There is **no dependency to a specific JavaScript IDE**. The IDE must be able to debug local Node process.

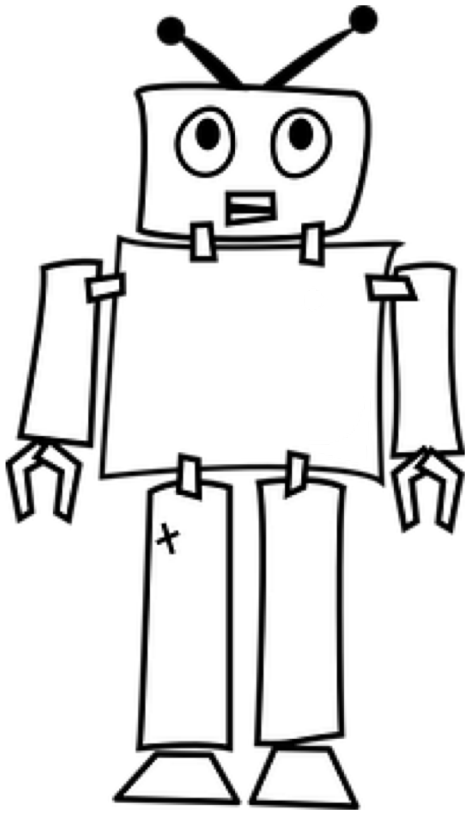
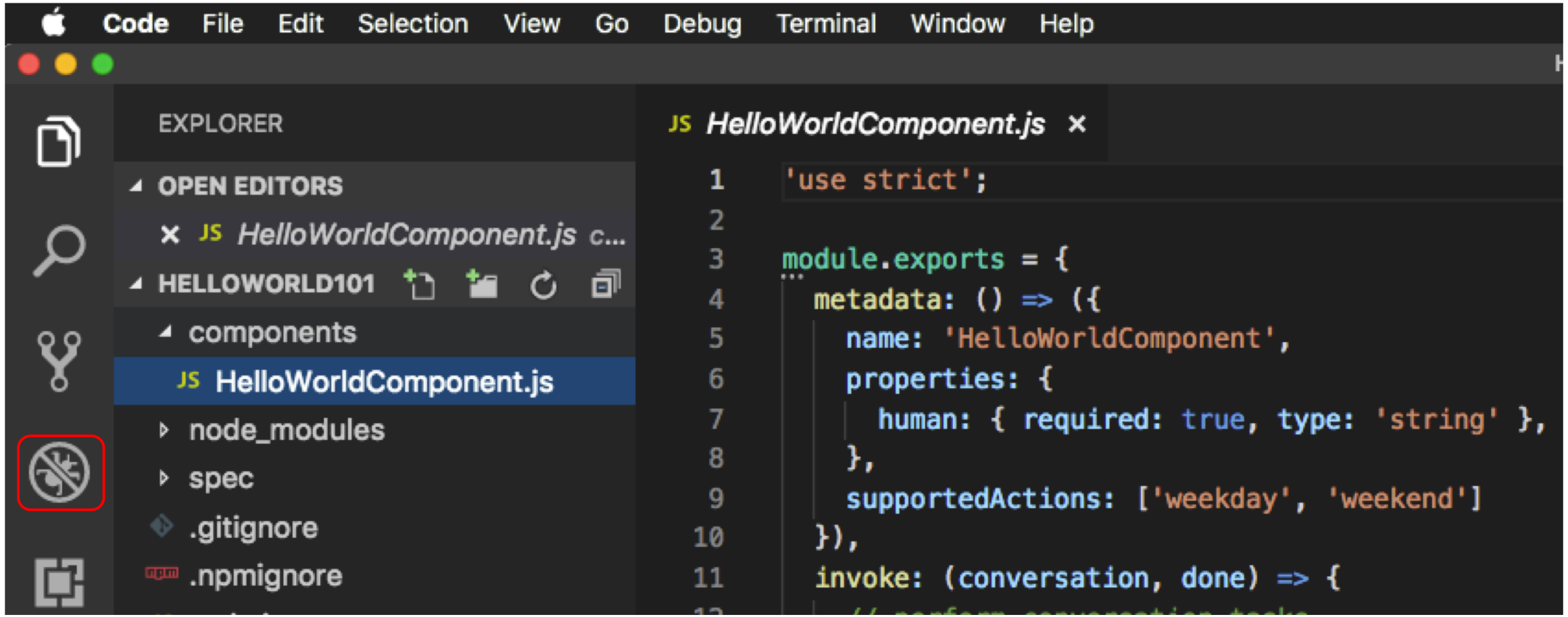
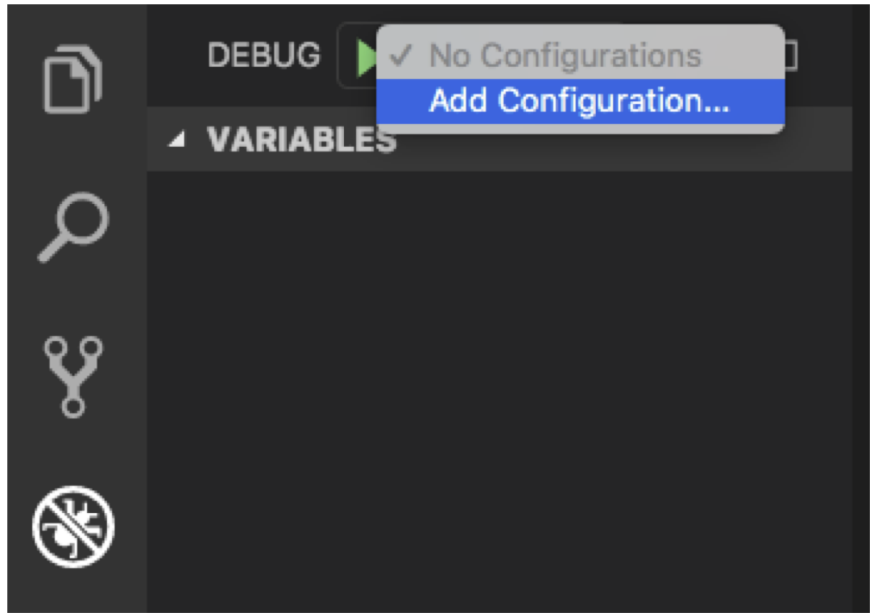


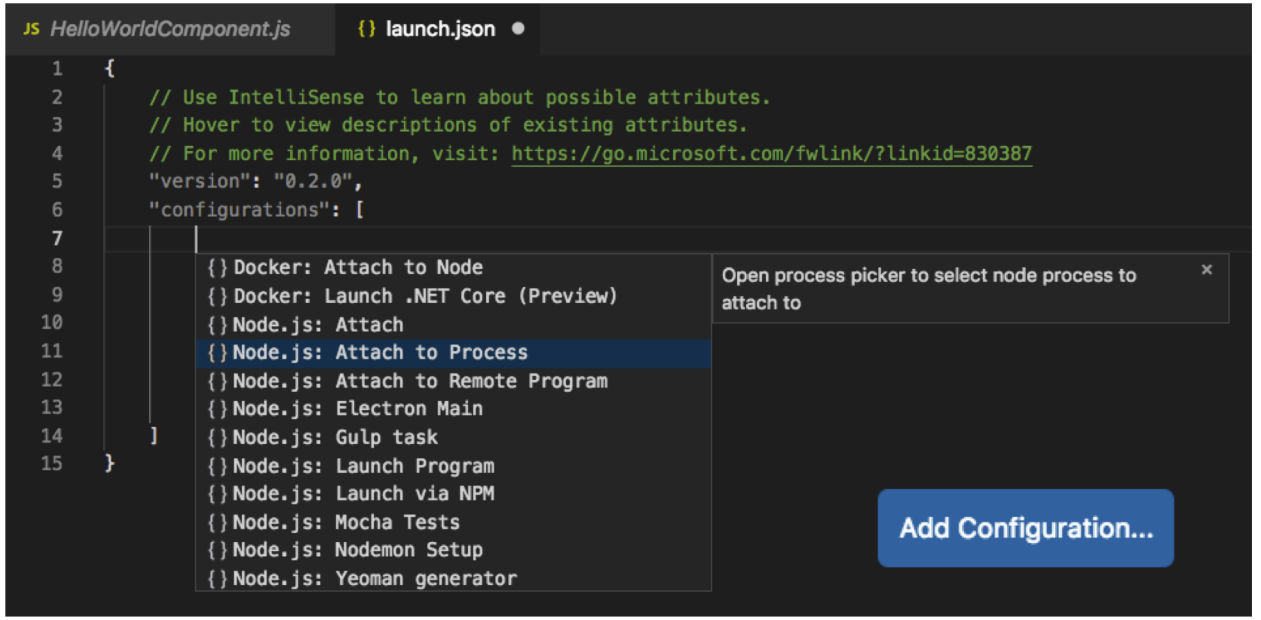
Image courtesy of pixabay.com

Open project in IDE and select debug option



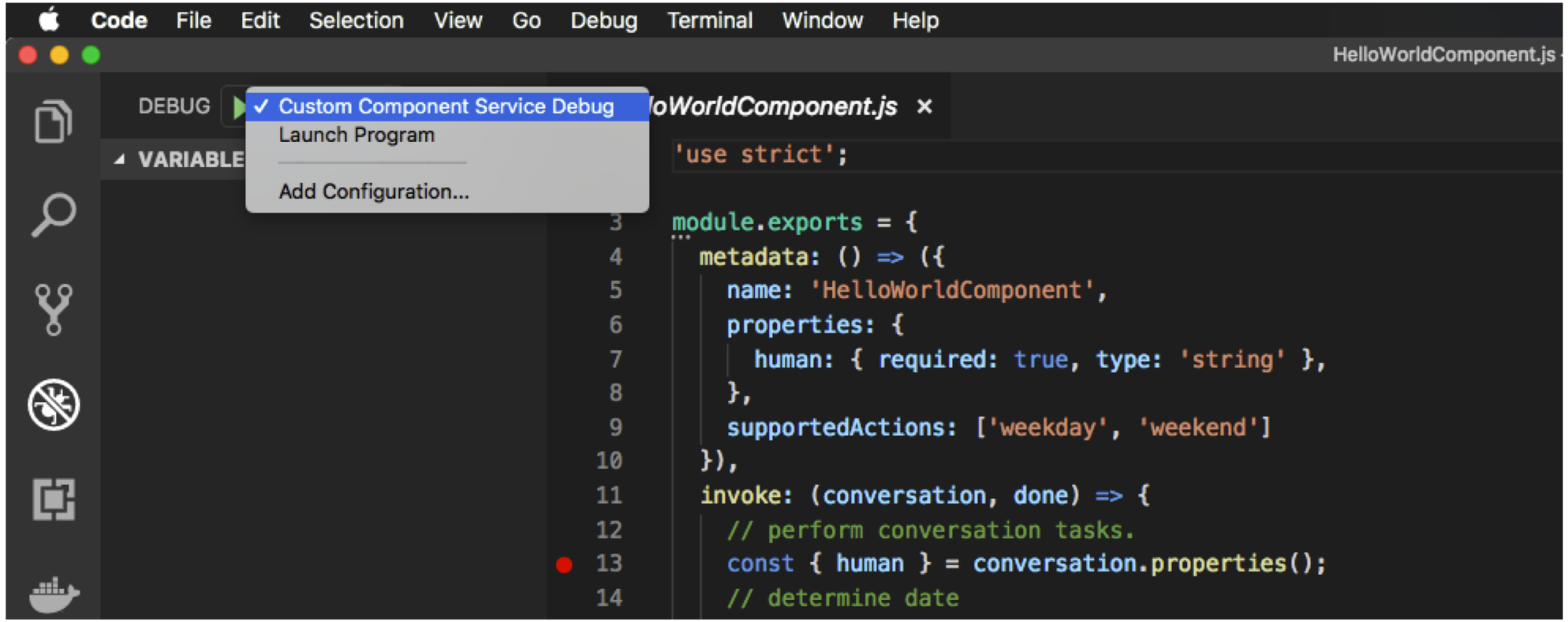
Create debugging configuration





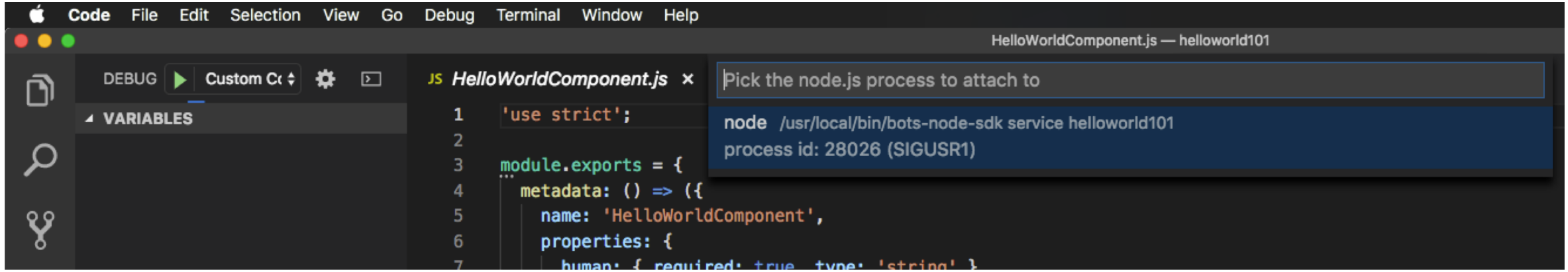
- Add a new debug configuration
  - Choose "Attach to Process" from "Add Configuration" button list
- Optionally, provide a custom name for the debug configuration

Set breakpoint and start debugging session



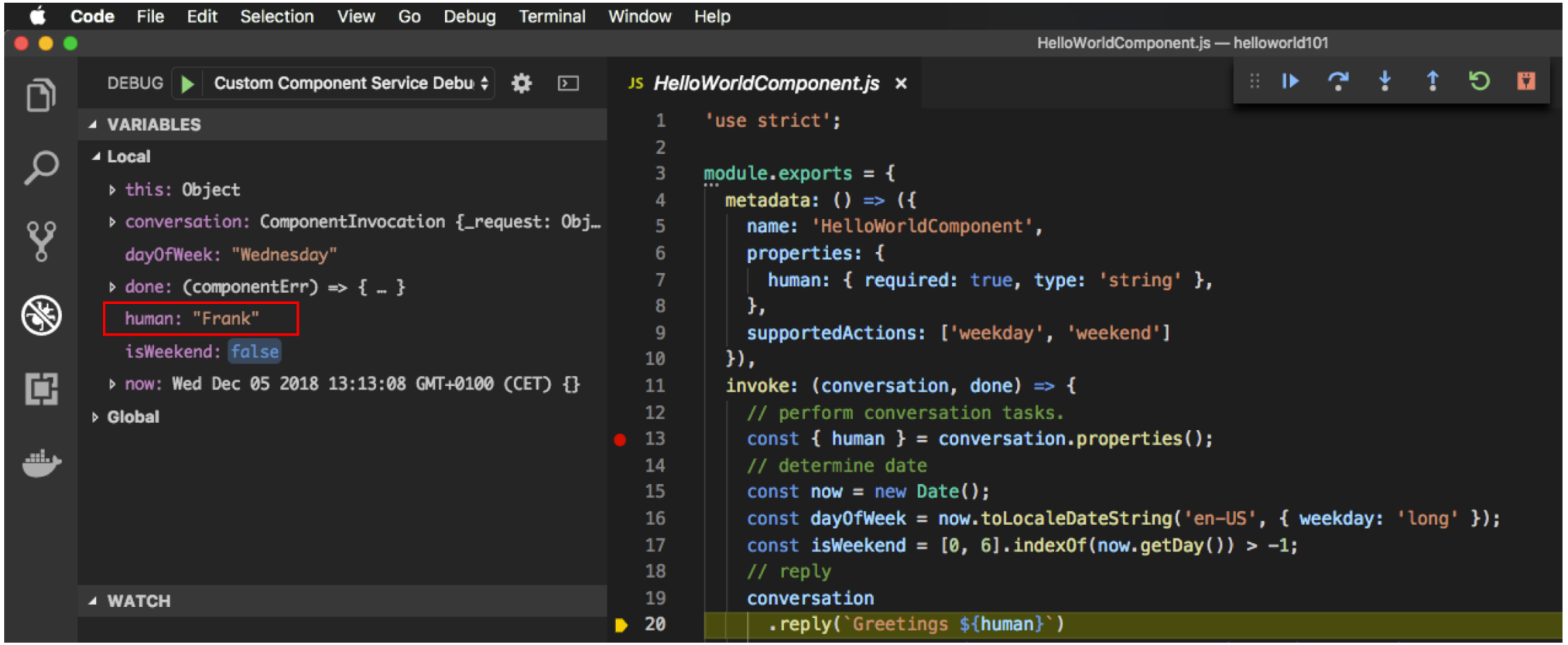
- Set break-points in the *invoke* function
- Select your custom debug configuration

Select process



- Press the green run icon
- Select the Node process from the list

Debug component service by running skill bot tester



Rerun the '**bots-node-sdk service**' command each time you change the component source code in a debug session for changes to take affect.

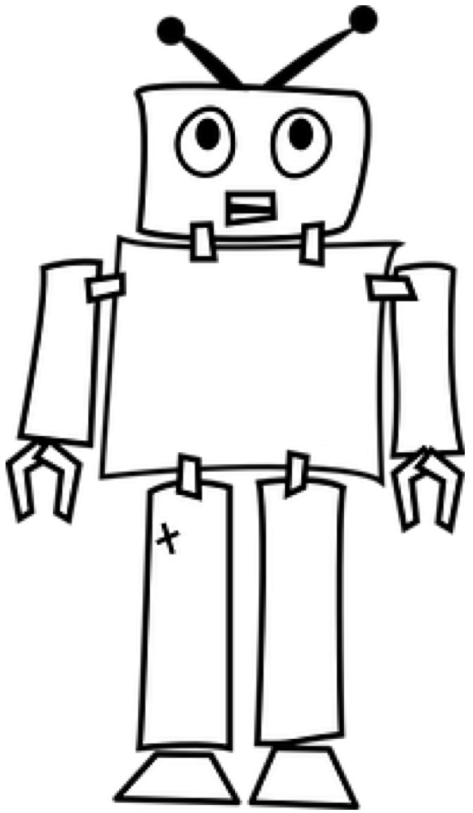
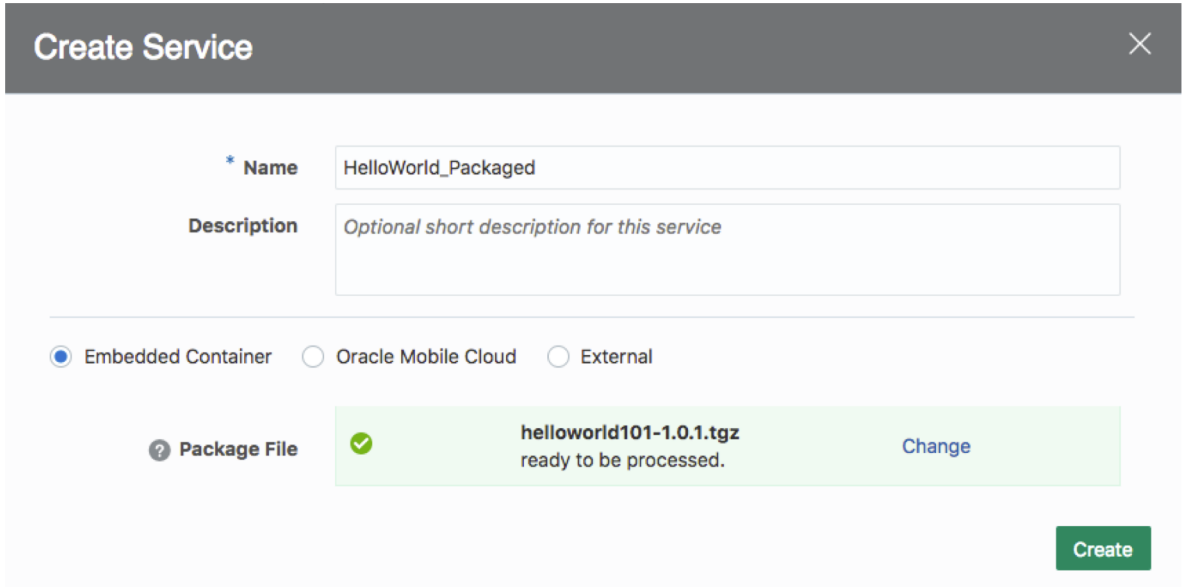


Image courtesy of pixabay.com

Custom component deployment to local container

- Create a .tgz file for local container deployment
  - Run *npm pack* in the component service project folder
- Delete or disable the current custom component service configuration
- Create new service configuration using packaged component service
  - Name different if you kept the debugging configuration



Integrated Cloud

Applications & Platform Services


ORACLE®

Oracle Digital Assistant Hands-On

TBD