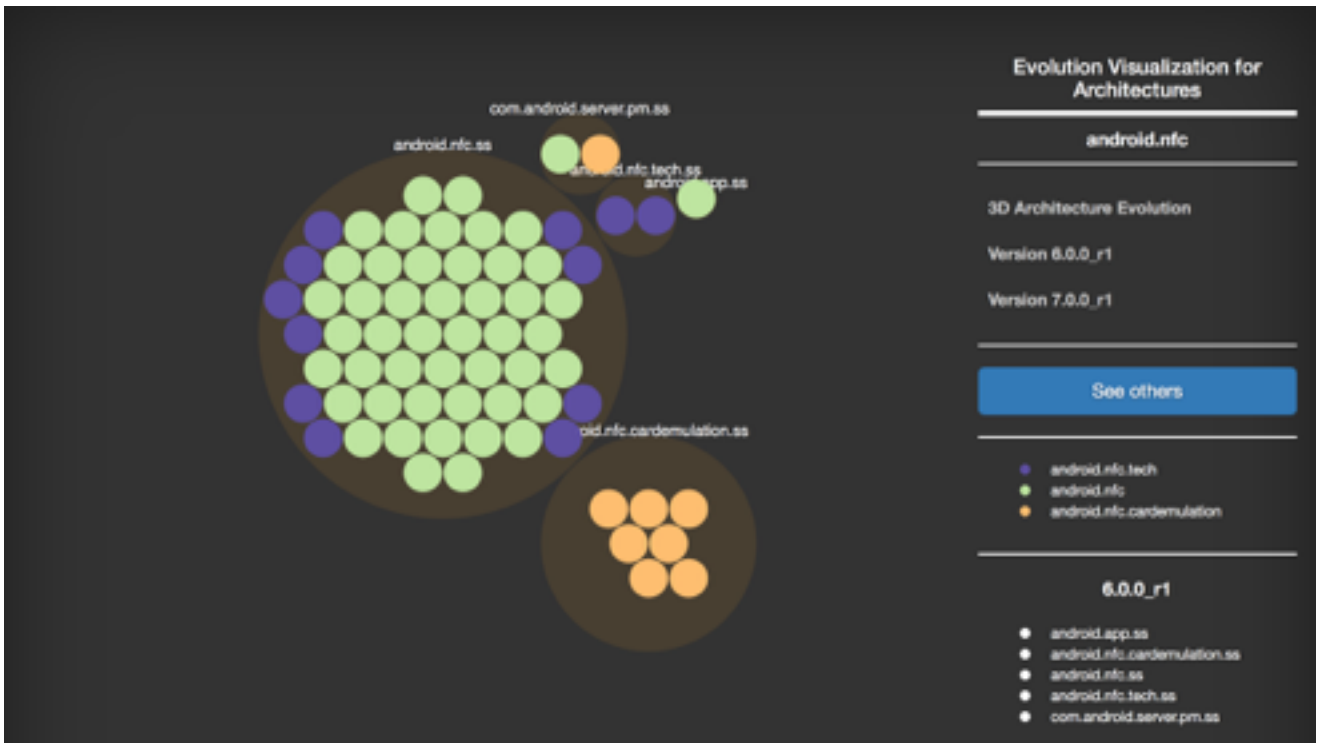
# EVA Manual

## EVA: A Tool for Visualizing Software Architectural Evolution




Computer Science Department, University of Southern California

*1st Edition*

# 1. Installing EVA

## 1.1 Requirements

The following is the operating system, execution environments, tools we have used to make sure EVA works as described in this manual. Different configurations may work, but have not beed tested.

### Operating Systems

- macOS Sierra, Version 10.12

### Browsers

- Chrome 65.0
- Safari 10.0

### Python

- Python 2.7

### Localhost

- MAMP (for mac)

### Display Size

- 27-Inch

Before you run EVA, you have to install below python packages.

- enum34 (1.1.6)
- ortools (6.6.4656)
- python-dateutil (2.4.2)

Since EVA's visualization is developed to be shown on a web-browser using HTML, CSS and Javascript, you need a localhost environment. For Mac, you can set up the localhost environment easily using MAMP (mac) or WAMP (windows). You can also move the source codes to your own server and access the visualization through the web browser.

## 1.2 Overview

EVA is written in Python, Javascript, HTML, and CSS.

1. EVA source code can be downloaded from the following Github repository:

   https://github.com/namdy0429/EVA-Huawei

2. Place the EVA project into the localhost folder.
3. Run EVA.py to generate the input data of the EVA visualization.
4. View the EVA's visualization with your browser.

# 2. Visualizing with EVA

## 2.1 Conventions used in this and the Following Sections

- In this section, Android 6.0.0 and Android 7.0.0 are used in an invocation example throughout this manual.
- Note that the following files have been provided in the folder hierarchy of the EVA project. If all examples in Section 3 are followed in sequence, the results should be exactly as indicated.

| Item | Location |
| --- | --- |
| Input Recovered Architectures | Data/Architecture/android/{recovery method}/{layer} |
| A List of Sub-systems under an Architectural Layer | Engine/config/ |

## 2.2 Processing Data

**Synopsis:**

```
python EVA.py –outputDir –recovery –arch1 –arch2 –layer
```

**Parameters:**

- `outputDir`
  This is the dictionary where all output files go, which can be the localhost EVA folder.
- `recovery`
  This is the recovery method that is used for architecture recovery.
- `arch1, arch2`
  These are .rsf format files resulting from running ARCADE. arch1 should be the older version, and arch2 is the newer version.
- `layer`
  Because of the scalability issue, a user will recover Android's architecture partially, based on each architectural layer. This is the name of the architectural layer that a user recovered.

  Also, you have to place a list of sub-systems under "config/", using this layer name.

### Example:

```
python EVA.py --outputDir /Applications/MAMP/htdocs/EVA/Front-end --
recovery acdc --arch1 android-6.0.0_r1_acdc_clustered.rsf --arch2
android-7.0.0_r1_acdc_clustered.rsf --layer framework
```

### Resulting Files:

```
EVA/Front-End/data/android/acdc/6.0.0_r1_7.0.0_r1_framework/
├── 6.0.0_r1_7.0.0_r1_android.accessibilityservice_processed_archs.json
├── 6.0.0_r1_7.0.0_r1_android.accounts_processed_archs.json
├── …
├── 6.0.0_r1_7.0.0_r1_com.android.server.webkit_processed_archs.json
└── 6.0.0_r1_7.0.0_r1_com.android.server.wm_processed_archs.json
EVA/Front-End/data/android/acdc/6.0.0_r1_framework/
├── 6.0.0_r1_android.accessibilityservice.ss_processed_archs.json
├── 6.0.0_r1_android.accounts.ss_processed_archs.json
├── …
├── 6.0.0_r1_com.android.server.wm.ss_processed_archs.json
└── 6.0.0_r1_com.google.android.util.ss_processed_archs.json
EVA/Front-End/data/android/acdc/7.0.0_r1_framework/
├── 7.0.0_r1_android.accessibilityservice.ss_processed_archs.json
├── 7.0.0_r1_android.accounts.ss_processed_archs.json
├── …
├── 7.0.0_r1_com.android.server.wm.ss_processed_archs.json
└── 7.0.0_r1_com.google.android.util.ss_processed_archs.json
```

## 2.3 Rendering Visualization

There are two ways of accessing EVA's visualization.

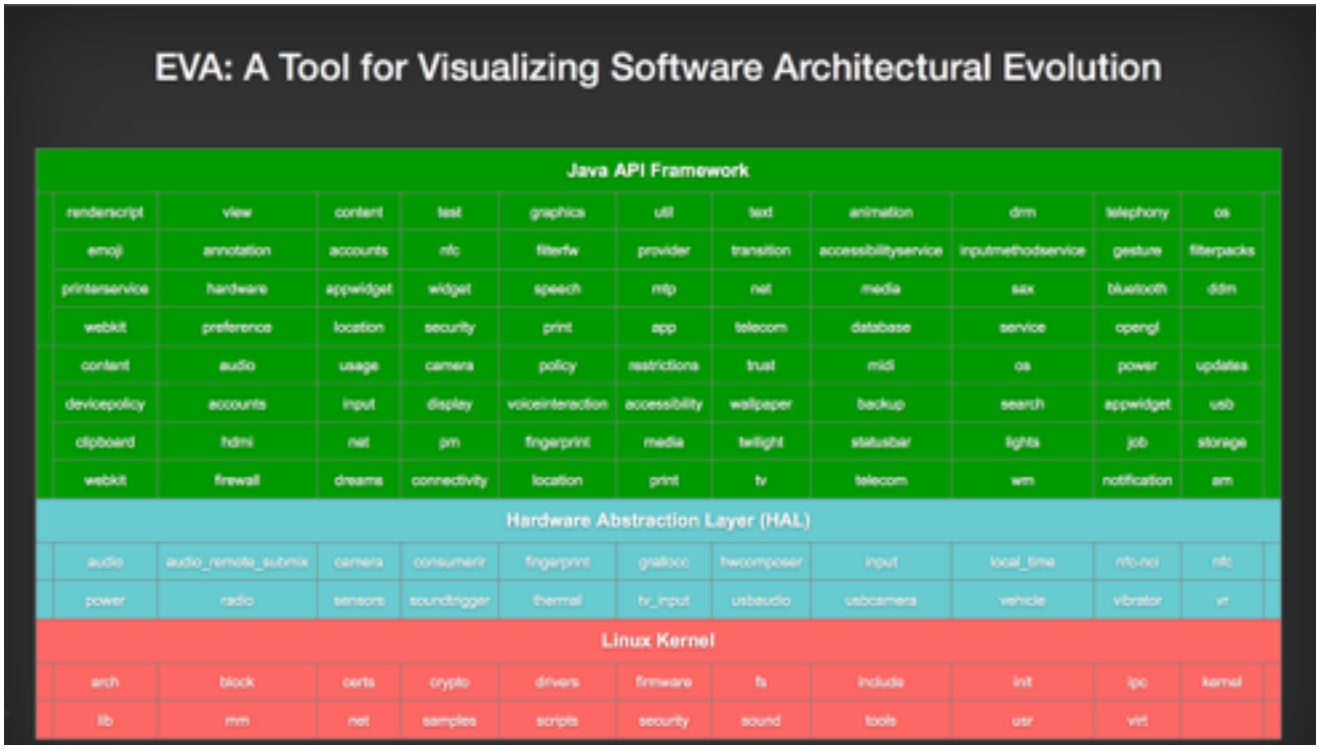


Fig 1. EVA index page.

## *(1) Start from the index page*

**URL:**

```
[localhost EVA path]/index.html?
&recovery=[recovery_method]&ver1=[version1]&ver2=[version2]&layer=[l
ayer]
```

**Parameters:**

- [localhost EVA path]: EVA Front-end directory within localhost folder.
- [recovery_method]: A recovery method used to recover the system architecture.
- [version1], [version2]: Two android versions to compare. Put the older version into [version1], and the newer version into [version2].
- [layer]: The target architectural layer for the visualization.

**Example:**

```
http://localhost:8888/EVA/Front-end/index.html?
&recovery=acdc&ver1=6.0.0_r1&ver2=7.0.0_r1&layer=framework
```

**Description:**

Each big colored block refers to one architectural layer of Android. Each small block within the architectural layer refers to a sub-system within the architectural layer.

When you click a sub-system, you can move to the corresponding visualization page.

## *(2) Query using GET*

**URL:**

```
[localhost EVA path]/index.html?
&recovery=[recovery_method]&ver1=[version1]&ver2=[version2]&layer=[l
ayer]&target_sub=[target_subsystem]
```

**Parameters:**

- [localhost EVA path]: EVA Front-end directory within localhost folder.

- [recovery_method]: A recovery method used to recover the system architecture.
- [version1], [version2]: Two android versions to compare. Put the older version into [version1], and the newer version into [version2].
- [layer]: The target architectural layer for the visualization.
- [target_subsystem]: The target sub-system for the visualization. The selected sub-system should be within the selected architectural layer.

Example:

```
http://localhost:8888/EVA/index_combined.html?
&recovery=acdc&ver1=6.0.0_r1&ver2=7.0.0_r1&layer=framework&target_su
b=android.media
```

## 2.4 Exploring System Architecture using EVA

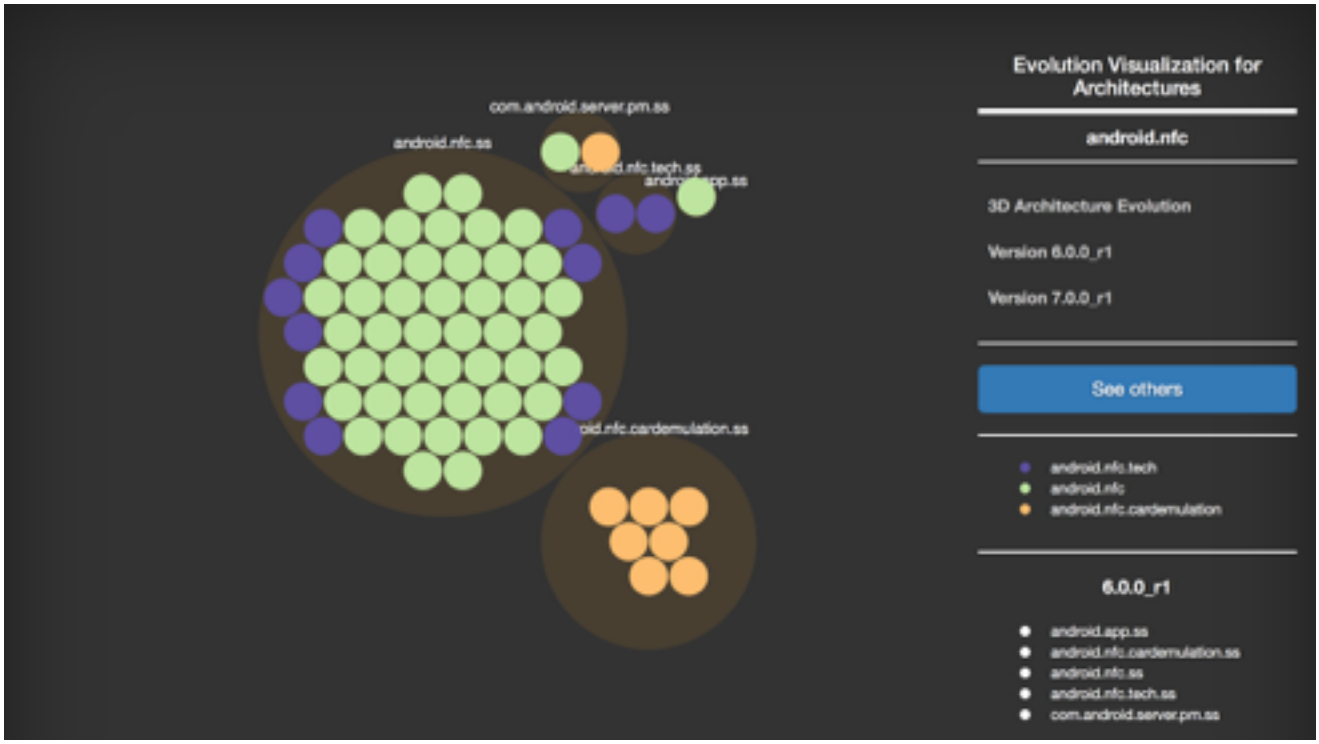



Fig 2. EVA Visualization page example. Visualizing android.nfc, which is a sub-system of Android Framework layer.

- EVA generates a visualization page for each sub-system. For example, Figure 2 shows the architecture of android.nfc in the Framework layer. When you combine all the sub-visualizations of one layer, it will show the whole architecture of the given architectural

layer. However, since Android is too big to visualize into a single page, we partitioned the visualization.

- Each bigger circle refers to a architectural component recovered using the recovery technique in ARCADE. The white texts above the circle is the name of each component.
- Each small dot refers to a code-level entities (e.g., class in java).
- The color of small dots shows the code-level's package information. For example, purple dots in Figure 2 are classes belong to android.nfc.tech package. From this color-coding, it is easy to see whether the code-level distributions corresponds to the system architecture.
- Click "See others" button to see the index table, and move to other sub-systems.
- You can see each component, without filtering for each sub-system, by clicking each component name on right list (e.g., android.app.ss).

# 3. Step-by-step

## 3.0 Intro

EVA uses architectures of two different versions as input, and help you to explore the differences between the two with the visualization. You have to run ARCADE for architecture recovery, and use resulting files as EVA's input.

Due to the huge size of Android, it is not easy to recover the whole system at once. Therefore, you will recover architectures for each architectural layer, such as Java API Framework or Hardware Abstraction Layer in Fig 3.

Fig 3. The Android software stack posted at https://developer.android.com/guide/platform/.

## 3.1 Check Localhost environment

Set up a localhost environment to test EVA on your local machine. For mac, you can easily set it up using MAMP. For windows, you can follow this video instruction, read this posting, or use WAMP or any program for easy set-up.

Open any browser (Chrome or Firefox are recommended) installed on your machine, and put below to your address bar.

```
http://localhost:8888
```

You should be able to see any page without getting 404 error when your localhost is set correctly.

## 3.2 Download source code for EVA https://softarch.usc.edu/~daye/EVA-Huawei/EVA.tar.gz

Download EVA source code, EVA.tar.gz, from above address, or clone from https://github.com/namdy0429/EVA-Huawei.

Extract EVA.tar.gz and place the whole EVA source code into your localhost folder. If you are clowning from Github, just move to below directory, and clone it.

- Mac (default):

  ```
  /Library/WebServer/Documents
  ```

- Mac (when you use MAMP) :

  ```
  /Applications/MAMP/htdocs
  ```

- Windows (default):

  ```
  C:/inetpub/wwwroot
  ```

After place the EVA folder at the localhost folder, you can see below.

```
Dayeui-MacBook-Pro:EVA namdy$ pwd
/Applications/MAMP/htdocs/EVA

Dayeui-MacBook-Pro:EVA namdy$ ls -la
drwxr-xr-x   8 namdy  admin      272 May 16 07:28 .
drwxrwxr-x  28 namdy  admin      952 May 16 11:13 ..
drwxr-xr-x   5 namdy  admin      170 May  1 06:03 Back-end
```

```
-rw-r--r--@  1 namdy  admin   304792 May  2 07:11 Documentation.pdf
drwxr-xr-x  10 namdy  admin      340 May 14 19:44 Front-end
```

## 3.3 Install necessary Python packages

For EVA's data-processing, you need to install below python packages. You can easily install by using pip.

- enum34 (1.1.6)
- ortools (6.6.4656)
- python-dateutil (2.4.2)

```
pip install enum34
pip install ortools
pip install python-dateutil
```

## 3.4 Place architecture files resulted from ARCADE

EVA uses .rsf files which are resulted from architecture recovery using ARCADE. You can find these files at ARCADE's clusters folder of each system's and recovery's result.

```
Example: {ARCADE_home}/subject_systems/Android/output/acdc/clusters
```

You will use two .rsf files, which are recovered architectures of different versions but of a same layer.

Example: recovered architectures of 6.0.0_r1 and 7.0.0_r1 Android Java API Framework

- `android-6.0.0_r1_acdc_clustered.rsf`
- `android-7.0.0_r1_acdc_clustered.rsf`

Place two .rsf files  that you want to compare into EVA's Architecture folder.

```
EVA/Back-end/Data/Architecture/{system name}/{recovery name}/{layer
name}
```

Example:

```
Dayeui-MacBook-Pro:acdc namdy$ pwd
/Applications/MAMP/htdocs/EVA/Back-end/Data/Architecture/android/
acdc/framework

Dayeui-MacBook-Pro:framework namdy$ ls -la
```

```
-rw-r--r--@ 1 namdy  admin  810461 Apr 10 20:24
android-6.0.0_r1_acdc_clustered.rsf
-rw-r--r--@ 1 namdy  admin  920580 Apr 10 20:47
android-7.0.0_r1_acdc_clustered.rsf
```

## 3.5 Add config files

List sub-systems in the target layer you want to visualize, and name the list as the layer name. Place the list at

```
{localhost}/EVA/Back-end/Engine/config/
```

For example, if you want to visualize sub-systems of Java Framework API layer in Fig 4,

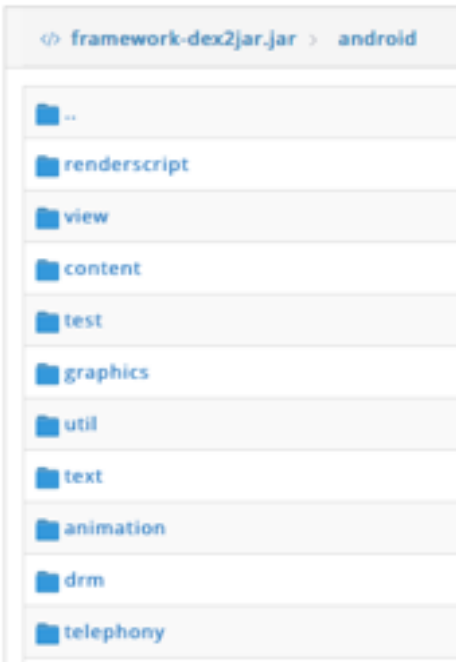



Fig 4. Sub-systems of Java API Framework

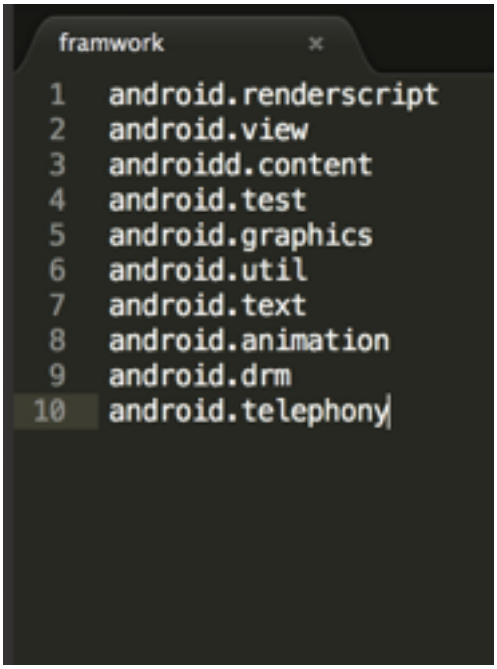



Fig 5. Framework config file

the config file should be Fig 5.

You can check example list at /Applications/MAMP/htdocs/EVA/Back-end/Engine/config.

## 3.6 Process data for EVA visualization

You can process two input .rsf files into a visualization input using EVA.py. First, Move to

```
[localhost]/EVA/Back-end/Engine
```

Run EVA.py with below parameters.

### Synopsis:

```
python EVA.py –outputDir –recovery –arch1 –arch2 –layer
```

### Parameters:

- `outputDir`

  This is the dictionary where all output files go, which will be the path to EVA's Front-end folder.

  `{localhost}/EVA/Front-end.`

- `recovery`

  This is the recovery method that is used for architecture recovery.

- `arch1, arch2`

  .rsf format files that you placed into EVA/Back-end/Data/Architecture/{system name}/{recovery name}/{layer name} at 3.4, which were resulted from ARCADE. arch1 should be the older version, and arch2 is the newer version.

- `layer`

  This is the name of the architectural layer that a user recovered. A config file with this name should be placed under "config" folder before running EVA.py.

### Example:

```
python EVA.py --outputDir /Applications/MAMP/htdocs/EVA/Front-end --
recovery acdc --arch1 android-6.0.0_r1_acdc_clustered.rsf --arch2
android-7.0.0_r1_acdc_clustered.rsf --layer framework
```

### Resulting Files:

```
EVA/Front-End/data/android/acdc/6.0.0_r1_7.0.0_r1_framework/
├── 6.0.0_r1_7.0.0_r1_android.accessibilityservice_processed_archs.json
├── 6.0.0_r1_7.0.0_r1_android.accounts_processed_archs.json
├── …
├── 6.0.0_r1_7.0.0_r1_com.android.server.webkit_processed_archs.json
└── 6.0.0_r1_7.0.0_r1_com.android.server.wm_processed_archs.json
EVA/Front-End/data/android/acdc/6.0.0_r1_framework/
├── 6.0.0_r1_android.accessibilityservice.ss_processed_archs.json
├── 6.0.0_r1_android.accounts.ss_processed_archs.json
├── …
├── 6.0.0_r1_com.android.server.wm.ss_processed_archs.json
└── 6.0.0_r1_com.google.android.util.ss_processed_archs.json
EVA/Front-End/data/android/acdc/7.0.0_r1_framework/
├── 7.0.0_r1_android.accessibilityservice.ss_processed_archs.json
├── 7.0.0_r1_android.accounts.ss_processed_archs.json
├── …
```

```
├── 7.0.0_r1_com.android.server.wm.ss_processed_archs.json
└── 7.0.0_r1_com.google.android.util.ss_processed_archs.json
```

## 3.7 Update index table

If added layers or sub-systems are not included in the provided index table, please add the cells for each sub-systems into index.html. In order to link the visualizations to this index table, each cell should be formed like below.

```
<td class="table_index {layer color class}“><a href="" layer=“{layer
name}” package=“{sub-system name}”>{sub-system name}</a></td>
```

For example,

```
<td class="table_index tg-3up7"><a href="" layer="hardware"
package="modules.power">modules.power</a></td>
```

## 3.8 Open visualizations using your browser

You can see the visualization using your web browser (Chrome or Firefox are recommended). Open your browser and put below url into your address bar.

**URL:**

```
{localhost}/EVA/Front-end/index.html?&recovery={recovery_method}
&ver1={version1}&ver2={version2}
```

**Parameters:**

- {localhost}: Localhost home directory.
- {recovery_method}: A recovery method used to recover the system architecture.
- {version1}, {version2}: Two android versions to compare. Put the older version into {version1}, and the newer version into {version2}.

**Example:**

```
http://localhost:8888/EVA/Front-end/index.html?
&recovery=acdc&ver1=6.0.0_r1&ver2=7.0.0_r1
```

You can directly move to a certain sub-system’s visualization by put below url into your address bar.

URL:

```
{localhost}/EVA/Front-end/index_combined.html?
&recovery=[recovery_method]&ver1=[version1]&ver2=[version2]&layer=[l
ayer]&target_sub=[target_subsystem]
```

Parameters:

- {localhost}: Localhost home directory.
- {recovery_method}: A recovery method used to recover the system architecture.
- {version1}, {version2}: Two android versions to compare. Put the older version into {version1}, and the newer version into {version2}.
- {target_subsystem}: The target sub-system for the visualization. The selected sub-system should be within the selected architectural layer.

Example:

```
http://localhost:8888/EVA/index_combined.html?
&recovery=acdc&ver1=6.0.0_r1&ver2=7.0.0_r1&layer=framework&target_su
b=android.media
```