

# Flutter demo apps on AGL

Lorenzo Tilve

AGL Virtual F2F 19 - 20 October 2021



# Flutter applications into AGL

- Goal of this presentation is to describe how to start testing flutter applications integration into the AGL demo platform.
- Multiple resources to start creating flutter apps from the scratch or using existing samples:
  - <https://flutter.dev/docs/get-started/codelab>
  - <https://flutter.dev/docs/development/ui/layout/tutorial>
  - <https://gallery.flutter.dev/>

# Flutter apps and WAM

- Flutter apps can be built as web bundles, and displayed and run using the Web Application Manager into the HTML homescreen.
- Packaged and tested together with other web applications in the HTML-only UI.

<https://confluence.automotivelinux.org/display/MAIN/Flutter+apps+on+AGL+with+tl>

- More specific details on how to build the HTML UI demo image:

<https://www.youtube.com/watch?v=quM6DXOAVvM>

## Native support on AGL

- Native flutter embedder for Wayland with ivi-homescreen presented by Toyota:

- 

<https://confluence.automotivelinux.org/pages/viewpage.action?pageId=49155374>

## Building flutter web bundles

```
# Develop the app and package it for web
# adding the corresponding config.xml
flutter build web
cd build/web
zip -r flutter-webapp.wgt

# For installing into Qemu
scp -P2222 *wgt root@localhost:
qemux86-64$ afm-util install flutter-webapp.wgt
```

# Application framework integration

- API integration with the AppFramework bindings for the webapps was done with JS interfaces:
  - <https://github.com/AGL-web-applications/agl-js-api/tree/master/src>
- In flutter we are now using an AGL/JS dart wrapper:
  - <https://github.com/rogerzanoni/agljs.dart>
- Imports `AGLJS` dependency from the repository into the `pubspec.yaml`

# Application framework integration (JS)

<https://github.com/AGL-web-applications/agl-js-api/blob/master/src/audiomixer.js>

```
export function list_controls() {
    return api_call("audiomixer/list_controls", {});
}
export function set_volume(control, value) {
    return api_call("audiomixer/volume", {
        control: control,
        value: value
    });
}
export function on_volume_changed(handler) {
    return api_subscribe("audiomixer/subscribe", {
        event: "volume_changed"
    }, handler);
}
```

# Application framework integration (Dart)

[https://github.com/rogerzanoni/agl-flutter-web-listview/blob/audiomixer\\_test/lib/main.dart](https://github.com/rogerzanoni/agl-flutter-web-listview/blob/audiomixer_test/lib/main.dart)

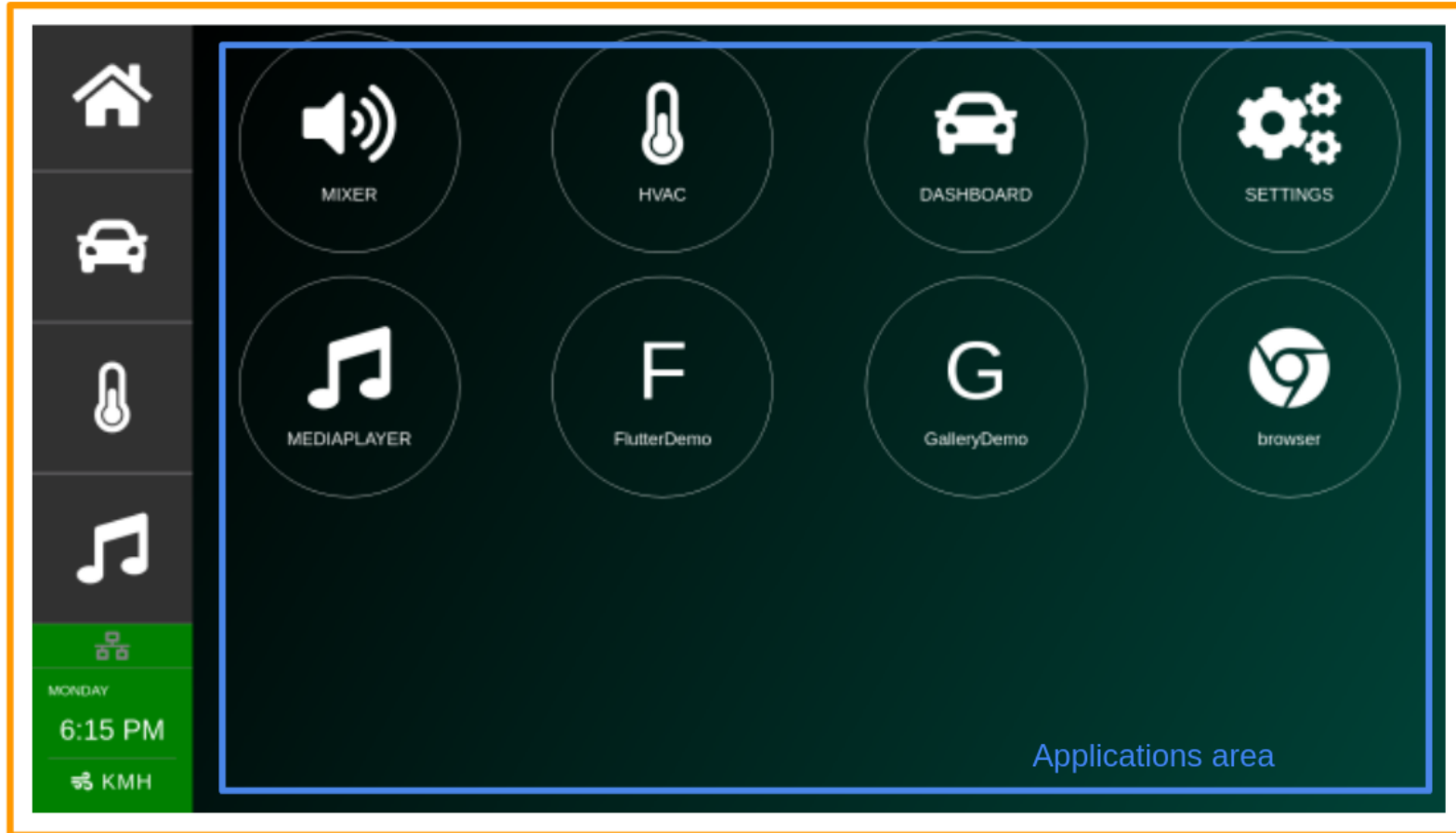
```
void _retrieveControls() {
  Future<List<AfmMixerControl>> runnablesFuture =
    promiseToFuture<List<AfmMixerControl>>(AGLJS.audiomixer.list_controls);
  runnablesFuture.then((List<AfmMixerControl> controls) =>
    _resetVolume(controls));
}

void _resetVolume(List<AfmMixerControl> controls) {
  controls.forEach((mixerControl) =>
    AGLJS.audiomixer.set_volume(mixerControl.control, 0));
}

void _registerAudiomixerCallbacks() {
  AGLJS.audiomixer.on_volume_changed(allowInterop((data) => _retrieveAp
}
```



# AGL web homescreen structure



Homescreen area



Iyalia

# AGL web homescreen structure

The screenshot displays the AGL web application's home screen. On the left is a vertical navigation sidebar with icons for Home, Car, Thermostat, Music, and a green bar at the bottom showing the date and time (MONDAY 6:19 PM) and a speed unit (KMH). The top navigation bar is purple and contains a search icon, a 'FLY' button, and 'SLEEP' and 'EAT' options. Below this are four search filters: 'Travelers', 'Choose Origin', 'Choose ...', and 'Select Dates'. The main content area is titled 'Explore Flights by Destination' and features a grid of six destination cards. Each card includes a representative image, the destination name, and flight details.

Destination	Flight Details
Big Sur, United States	Nonstop · 13h 30m
Khumbu Valley, Nepal	Nonstop · 5h 16m
Machu Picchu, Peru	2 stops · 19h 40m
Malé, Maldives	Nonstop · 8h 24m
Mexico City, Mexico	Nonstop · 5h 24m
Vitznau, Switzerland	1 stop · 14h 12m

# AGL web homescreen structure

## Ongoing work

- Continue the integration with the AppFramework
  - Adapt AGL-JS api calls into `agljsdart` wrapper.
- Migration of `flutter-launcher`, `flutter-hvac`, `flutter-mixer`, etc.
- Test embedding web applications with flutter `WebView` plugin.
- Review the layout of the homescreen and overall demo apps look and feel.

## Open questions

- Integration with native ivi-homescreen.
- Running multiple flutter applications homescreen and launcher.
- Adapt dart wrapper to the new App Framework bindings.
- Handling of application life cycle.

