ENABLING RICH WEB APPLICATIONS FOR IN-VEHICLE INFOTAINMENT.

USING THE WEBINOS PLATFORM INSIDE THE AUTOMOTIVE DOMAIN.
AGENDA.

Motivation
Why Web & Automotive?

Background
What Is webinos?

Our Approach
Vehicle Data for Web Apps

Live Demo
webinos Automotive Apps

Open Questions
The Road Ahead of Us

Lessons learned
Take Home Message
MOTIVATION: CURRENT LANDSCAPE OF IVI SYSTEMS.

- Long **time-to-market** for in-car infotainment applications and services

- **Highly fragmented landscape** for In-Vehicle Infotainment (IVI) systems

- Customer **demand for more personalization** options on IVI systems (seamless use of services across different devices)

- Difficult to **attract third-party developers** for IVI systems
MOTIVATION: UBIQUITY OF THE WEB AND THE BROWSER.

- General **trend towards Web- and browser-based services and applications** on smartphones, tablets and desktops

- **Highly standardized** runtime environment for application and services

- Large developer base

**Deduction:**
The Browser is the preferred candidate for a runtime environment on IVI systems.
BACKGROUND: WHAT IS WEBINOS?

- An open, cross-device and browser-based application platform
- Research project funded by the European Commision
- 30 partners
  - Device manufactures
  - Automotive manufactures
  - Mobile network operators
  - Small and medium businesses
  - Research Institutes
  - Standardization bodies
BACKGROUND: SAY HELLO TO WEBINOS!
BACKGROUND: SAY HELLO TO WEBINOS!
KEY QUESTION:
HOW TO GET ACCESS TO VEHICLE DATA?
OUR APIS: DESIGN APPROACH.

- Use data from other standardized APIs instead of duplicating them
- Asynchronous model to retrieve dynamic vehicle data
- Use case driven grouping of data properties to simplify access for web apps
- Minimize resource overhead for providing vehicle data

Two ‘automotive’ specific APIs

- Vehicle API (for retrieving vehicle specific data)
- Navigation API (for interacting with navigation system)
VEHICLE API: GENERAL CONCEPT.

- Soley **read access** to vehicle data (at this time)
- Distinction between **static** (e.g. transmission type) and **dynamic** (e.g. gear) **vehicle data**
- Developer registers **callback handlers** for retrieving dynamic vehicle data
- Dynamic data can be **requested once** (**get**) or can be **monitored** (**addEventListener**)  
- API is provided as part of `window.webinos.vehicle` object
VEHICLE API: WHICH DYNAMIC DATA IS PROVIDED?

```javascript
vehicle.get('tripcomputer', tcHandler); // one-time request
vehicle.addEventListener('tripcomputer', tcHandler); // monitor tripcomputer

function tcHandler(data){
    console.log('Average consumption': + data.averageConsumption);
    console.log('Average speed': + data.averageSpeed);
    console.log('Average trip speed': + data.tripSpeed);
    // ...
}
```
VEHICLE API: WHICH STATIC DATA IS PROVIDED?

- brand (string)
- model (string)
- year (string)
- fuel (enum)
- hybrid (enum)
- steeringwheel
- transmission
NAVIGATION API: GENERAL CONCEPT.

- simple API to interact with navigation software
- API is provided as part of the `window.webinos.navigation` object
- query for Point-of-Interests within a specified area (findDestination)
- Set the next destination of the navigation system (requestGuidance)
- Retrieve status of the navigation system (getStatus)
NAVIGATION API: CODE EXAMPLE.

Find a destination

```javascript
var destinations = new Array();

webinos.navigation.findDestination("BMW Welt", destinationCB, errorCB);

function destinationCB(pois){
  if(destinations.length > 0){
    navigateTo(destinations[0])
  } else{
    console.log("No POI found");
  }
}
```

Request guidance

```javascript
function navigateTo(destination){
  webinos.navigation.requestGuidance(
    destination, false, navigationHandler);
}

// callback handler for guidance events
var navigationHandler = {
  onRequest: function(id, poi){
    console.log('Guidance set to ' + poi.name);
  },
  onReach: function(id, poi){
    console.log(poi.name + ' reached.');
  },
  onCancel: function(id, poi){
    console.log('Guidance to ' + poi.name + ' is cancelled.');
  }
}
```
NAVIGATION API: CODE EXAMPLE.

Find a destination

```javascript
var destinations = new Array();
webinos.navigation.findDestination("BMW Welt", destinationCB, errorCB);

function destinationCB(pois){
  if(destinations.length > 0){
    navigateTo(destinations[0])
  } else{
    console.log("No POI found");
  }
}
```

Request guidance

```javascript
function navigateTo(destination){
  webinos.navigation.requestGuidance(
    destination, false, navigationHandler);
}
```

```
//callback handler for guidance events
var navigationHandler = {
  onRequest: function(id, poi){
    console.log(‘Guidance set to’ + poi.name);
  },
  onReach: function(id, poi){
    console.log(poi.name + ‘ reached.’);
  },
  onCancel: function(id, poi){
    console.log(‘Guidance to ’ + poi.name + ‘ is cancelled.’);
  }
}
```
NAVIGATION API: CODE EXAMPLE.

Find a destination

```javascript
var destinations = new Array();

webinos.navigation.findDestination("BMW Welt", destinationCB, errorCB);

function destinationCB(pois){
  if(destinations.length > 0){
    navigateTo(destinations[0])
  } else{
    console.log("No POI found");
  }
}
```

Request guidance

```javascript
function navigateTo(destination){
  webinos.navigation.requestGuidance(
    destination, false, navigationHandler);
}

//callback handler for guidance events
var navigationHandler = {
  onRequest: function(id, poi){
    console.log('Guidance set to ' + poi.name);
  },
  onReach: function(id, poi){
    console.log(poi.name + ' reached.');
  },
  onCancel: function(id, poi){
    console.log('Guidance to ' + poi.name + ' is cancelled.');
  }
}
```
NAVIGATION API: CODE EXAMPLE.

Find a destination

```javascript
var destinations = new Array();
webinos.navigation.findDestination("BMW Welt", destinationCB, errorCB);

function destinationCB(pois){
    if(destinations.length > 0){
        navigateTo(destinations[0])
    } else{
        console.log("No POI found");
    }
}
```

Request guidance

```javascript
function navigateTo(destination){
    webinos.navigation.requestGuidance(
        destination, false, navigationHandler);
}

// callback handler for guidance events
var navigationHandler = {
    onRequest: function(id, poi){
        console.log('Guidance set to ' + poi.name);
    },
    onReach: function(id, poi){
        console.log(poi.name + ' reached. ');
    },
    onCancel: function(id, poi){
        console.log('Guidance to ' + poi.name + ' is cancelled. ');
    }
}
```
HOW TO INTEGRATE WEBINOS INTO THE VEHICLE.
WEBINOS CORE CONCEPT: SEPARATING APPLICATION RUNTIME FROM DATA ACCESS.

IVI-System

Application Runtime

<table>
<thead>
<tr>
<th>Web App</th>
<th>Web App</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>webinos.js (API Proxy)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

WebSocket Connection

jQuery-RPC 2.0

Vehicle Data Provider

Personal Zone Proxy

<table>
<thead>
<tr>
<th>Message Handler</th>
</tr>
</thead>
<tbody>
<tr>
<td>API Manager</td>
</tr>
</tbody>
</table>

Vehicle Access Manager

CAN MOST

Kernel

| MOST Driver |
| CAN Driver  |
| ...        |

Hardware

| MOST | CAN | ... |

Smartphone Tablet PC Home Media

globally accessible: pzh.webinos.org

Vehicle data access in webinos, BMW Group Research and Technology, November 2012
WEBINOS VEHICLE EVALUATION PLATFORM.

Basic infrastructure

Demonstrator box
WEBINOS VEHICLE DEMONSTRATION PLATFORM.
Browser-based trip computer for in-car head units
- Pure web technology (HTML, CSS, JavaScript, Canvas)
  = HTML5 app
- webinos API for accessing vehicle data

Seamless trip planning on desktop, smartphone, and IVI-System
- Create travel itineraries
- Manage points of interest
- Data is automatically synchronized between the devices within a personal zone.
THE ROAD AHEAD OF US.
THE ROAD AHEAD OF US.

- How are we going to **control the access** to the vehicle bus?
- How to **enable write access** in conjunction with access control?
- **Adapting Web applications** to be safely used inside the vehicle
  - Handling different input controls
  - Adjusting graphical user interface
- Can we agree on a **common interface for vehicle data**?
TAKE HOME MESSAGE.

- **Proof-of-concept** for exposing vehicle data to Web applications
- **Read only** for vehicle data
- **Security** and **Safety** need to be solved properly
- **UI Constraints** need to be addressed