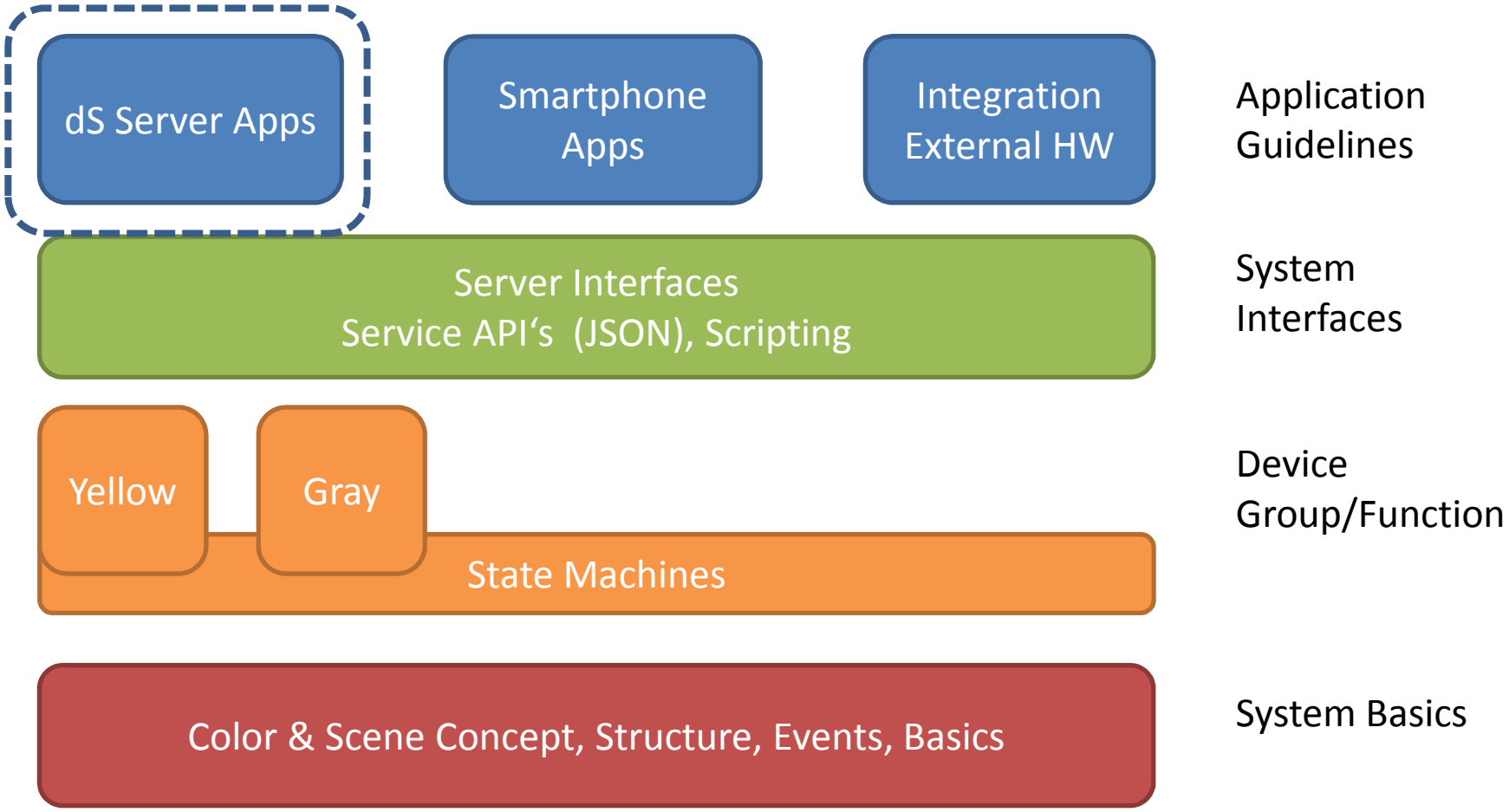
digitalSTROM

Developer Day
dS Apps

Roman Köhler

Sept 20th 2012

System Documentation



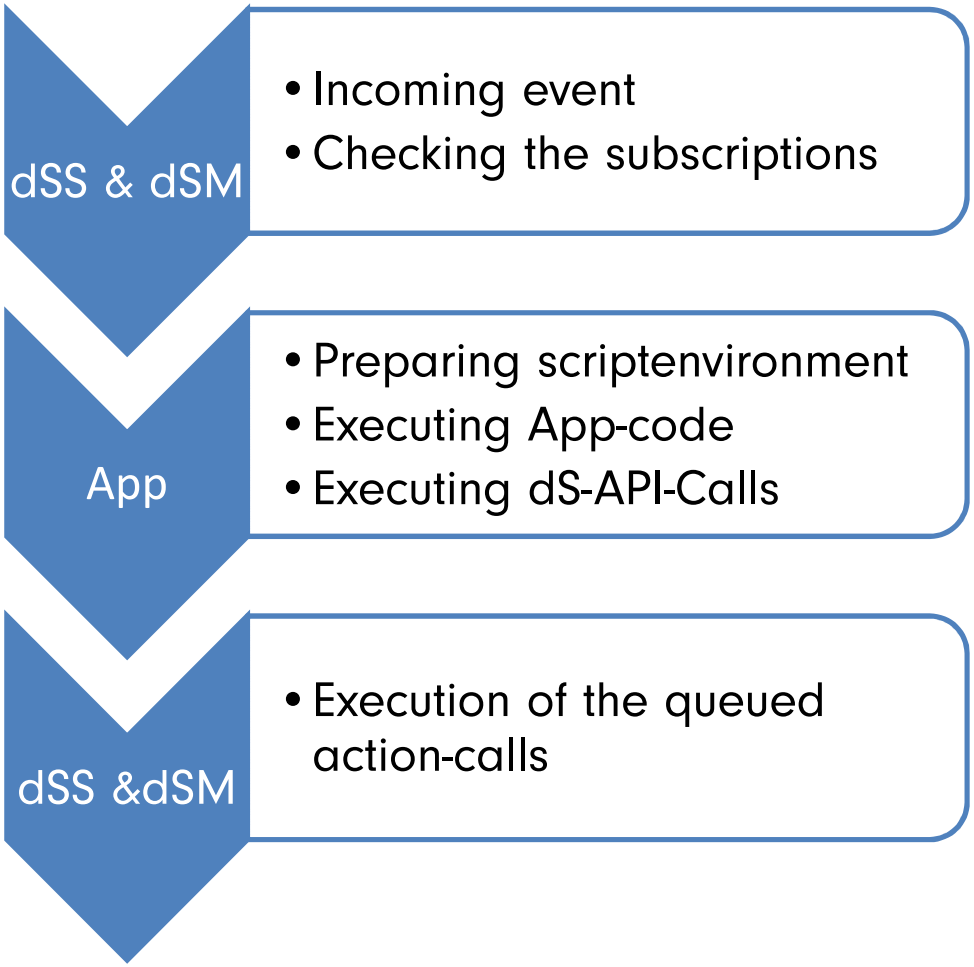
digitalSTROM

Topics

- Basics
  - Mechanism of a dS-App
  - Events and Subscriptions
  - Coding
  - Datasources
  - UI
- Practical Example
- Extended und included services
- Best practices

digitalSTROM

Basics – Mechanism of a dS-App



- Events can be originated from dS-Device, dSM-Statemaschine or dSS-Events
- Matching to Subscription is defined via XML-file of the dS-App
- dSS & dSM will execute the API-Calls, which might cause new Events

digitalSTROM

Basics - Eventsubscription

```
<subscription
    event-name=„callScene" handler-name="javascript">
    <parameter>
      <parameter name="filename1">
              myFile.js
      </parameter>
      <parameter name="filename2">
              myOtherFile.js
      </parameter>
     <parameter name="script_id">
              myApp
     </parameter>
</subscription>
```

- Typical events:

callScene, modelReady, running, highlevelevent, sendmail, buttonClick, deviceSensorEvent

- Filtering in the subscription is possible for some events, but such filtering is static and can not be changed by End-User settings

digitalSTROM

Basics - Coding

- Runs on a raised event
- Get Event-information via **raisedEvent** global variable
- dS-specific Extension:
  - Issue dS-system-commands (call a Scene, get a deviceparameter, raise a new Event)
  - Accessing internal datasources (Metering-Database, Propertytree)
- General Extension:
  - simple TCP-Socket-Support
  - CURL-Functions for advanced HTTP-Requests.

Language: ECMAScript 5

Engine: SpiderMonkey 1.8.5 including native JSON-Support

digitalSTROM

Basics- Datasources

- Property-Tree
  - Hierarchical organized tree data structure
  - Each node have a value or childs, and attributes if it should be serialized and read/writeable
  - Dynamical datastructure, will be build up each restart, parts are serialized.
  - Contains all available information of the running system
- Metering Database
  - automatic filled by a background-process from cycling dSM-queries
  - Values are readable, but not writable for Apps

- Apps can only save and restore data, which are put in Propertytree /scripts/<script_id>.
- Apps could hijack data from other apps, be aware of security-issues.

digitalSTROM

Basics UI

- Each App can deliver some HTTP-Pages, which will be hosted in the dSS-Webserver
- dSS provide JSON-Request for:
  - Get information of the structure of the installation
  - Accessing the Propertytree
  - Querying the metering database
  - Execute dSS-Commands like Event-Raising, Scene-Calling and Device-Configuration

- No server-side scripting possible, only the JSON-Interface of the dSS.
- No POST-Requests, only GET-Requests
- HTML-Files from a App are placed in a subdirectory named like the App, however, for systemcalls, this HTML-Files are not bound to a specific app

digitalSTROM

Practical Example

Accessing a webside and react on answer

```
<?xml version="1.0"?>
<subscriptions version="1">
  <subscription event-name="wetter-url" handler-
    name="javascript">
     <parameter name="filename1">
       /usr/share/dss/add-ons/wetter-url/wetter-url.js
      </parameter>
      <parameter name="script_id">wetter-
url</parameter>
  </subscription>
</subscriptions>
```

- Actual available Apps can call a URL to notify a remove server. The Answer is not in any app be evaluated
- CURL might block the dSS if the URL is not avaible

digitalSTROM

```
if (raisedEvent.name=='wetter-url') {
    var h = new HTTP( );
    var data =
h.get('http://www.wunderground.com/global/stations/06660.html');
    var sString=data.body;
    // do some (primitive) parsing
    sString=sString.substr(sString.indexOf('tempActual'));
    sString=sString.substr(sString.indexOf('<span class="b">')+16,100);
    sString=sString.substr(0,sString.indexOf('</span>'));
    var dTemperatur=parseFloat(sString);  // 20.09.2012 : 7.3 C
     // react on data -> Raise a Custom Event if < 15 C (id:1025)
     if (dTemperatur<15) {
         (new Event('highlevelevent',{id:1024})).raise();
     }
}
```

digitalSTROM

Extended und included services

- Custom Events/Highlevelevents
  - Special Events, which are administrated mainly be system-addons, can be executed by a simple event-raise.
- Action
  - Structure of well-defined Property-Nodes, which defines a sequence of actions and can be “executed” by a simple event-raise
- Trigger
  - Structure of well-defined Property-Nodes, which defines a dynamic subscription definition with explicit filtering on system-events
- Conditions
  - Structure of well-defined Property-Nodes, generalizes Checks for actual system-conditions

This services are result of the system-addon development and might be extended. It is advisable to take care of version numbers

digitalSTROM

Best Practices

Does

- Initialize a App on a model-ready event
- Recurring cyclic Events can be configured via subscription and a iCal-Timed Event
- Use Try/catch
- Communicate between App and UI via Event-Raise
- Use the JSON-Call **query** for accessing the property-tree

Don'ts

- Refrain from using setTimeout as far as possible
- Refrain from using property-listeners as far as possible
- Limit the script-execution time; break a task in subtasks
- Don't manipulate propertytree nodes from a foreign app directly
- Do not abuse the Logger-Utility

- When using to much memory, a script can be stopped anytime
- Calls to devices might take some time (0.5-2 seconds)
- Calls to devices are queued, however this queue is very limited

Please be nice to your dSS-11 Hardware, it has limited ressources

digitalSTROM