dSS JSON API und JavaScript

Stromverbrauchsanzeige im Browser mit 5 Zeilen Javacript

Andreas Stricker

futureLAB AG

2011-01-27

futureLAB

Übersicht

Zielpublikum

- Sie haben noch nicht mit der dSS JSON API gearbeitet
- Es sind nur geringe JavaScript Kenntnisse notwendig
- Einstieg zum Lernen und Entdecken der Möglichkeiten

Übersicht

Andreas Stricker

- Angestellt bei futureLAB AG
- Arbeitet nicht an digitalSTROM Projekten
- Letzter Kontakt mit dSS: vor einem Jahr

Übersicht

Andreas Stricker

- Angestellt bei futureLAB AG
- Arbeitet nicht an digitalSTROM Projekten
- Letzter Kontakt mit dSS: vor einem Jahr

Übersicht

Andreas Stricker

- Angestellt bei futureLAB AG
- Arbeitet nicht an digitalSTROM Projekten
- Letzter Kontakt mit dSS: vor einem Jahr

Inhalt

Das JSON API

dSS JSON API

- Basis URL: `http://localhost:8080/json`
- digitalSTROM Ressourcen
  - apartment
  - zone
  - device
  - circuit
- Weitere Ressourcen
  - property
  - event
  - metering
  - ...

Das JSON API

dSS JSON API

- Basis URL: `http://localhost:8080/json`
- digitalSTROM Ressourcen
  - apartment
  - zone
  - device
  - circuit
- Weitere Ressourcen
  - property
  - event
  - metering
  - ...

Das JSON API

dSS JSON API

- Basis URL: `http://localhost:8080/json`
- digitalSTROM Ressourcen
  - apartment
  - zone
  - device
  - circuit
- Weitere Ressourcen
  - property
  - event
  - metering
  - ...

Das JSON API

RPC Interface

- Das letzte Element der URL ist ein Funktionsname
- Danach folgen die Parameter als normale URL Parameter
- Beispiel mit Devices:
  `https://localhost:8080/json/device/getName?dsid=350...`
- Funktionen werden auch über HTTP `GET` aufgerufen

Das JSON API

RPC Interface

- Das letzte Element der URL ist ein Funktionsname
- Danach folgen die Parameter als normale URL Parameter
- Beispiel mit Devices:
  `https://localhost:8080/json/device/getName?dsid=350...`
- Funktionen werden auch über HTTP `GET` aufgerufen

Das JSON API

RPC Interface

- Das letzte Element der URL ist ein Funktionsname
- Danach folgen die Parameter als normale URL Parameter
- Beispiel mit Devices:
  `https://localhost:8080/json/device/getName?dsid=350...`
- Funktionen werden auch über HTTP `GET` aufgerufen

Das JSON API

RPC Interface

- Das letzte Element der URL ist ein Funktionsname
- Danach folgen die Parameter als normale URL Parameter
- Beispiel mit Devices:
  `https://localhost:8080/json/device/getName?dsid=350...`
- Funktionen werden auch über HTTP `GET` aufgerufen

Das JSON API

Mit dem JSON API "spielen"

- Über die Kommandozeile mit `curl` oder `wget`
  - `curl -k 'https://localhost:8080/json/apartment/getName'`
  - `wget -qO - --no-check-certificate 'https://localhost:8080/json/apartment/getName'`
  - Möglicherweise muss noch
- Im Browser API URLs aufrufen (z.B. Firefox mit JSONovich oder Firebug Plugin)

Das JSON API

Mit dem JSON API "spielen"

- Über die Kommandozeile mit `curl` oder `wget`
  - `curl -k 'https://localhost:8080/json/apartment/getName'`
  - `wget -qO - --no-check-certificate 'https://localhost:8080/json/apartment/getName'`
  - Möglicherweise muss noch
- Im Browser API URLs aufrufen (z.B. Firefox mit JSONovich oder Firebug Plugin)

Das JSON API

Mit dem JSON API "spielen"

- Über die Kommandozeile mit `curl` oder `wget`
  - `curl -k 'https://localhost:8080/json/apartment/getName'`
  - `wget -qO - --no-check-certificate 'https://localhost:8080/json/apartment/getName'`
  - Möglicherweise muss noch
- Im Browser API URLs aufrufen (z.B. Firefox mit JSONovich oder Firebug Plugin)

Das JSON API

Mit dem JSON API "spielen"

- Über die Kommandozeile mit `curl` oder `wget`
  - `curl -k 'https://localhost:8080/json/apartment/getName'`
  - `wget -qO - --no-check-certificate 'https://localhost:8080/json/apartment/getName'`
  - Möglicherweise muss noch
- Im Browser API URLs aufrufen (z.B. Firefox mit JSONovich oder Firebug Plugin)

Das JSON API

Mit dem JSON API "spielen"

- Über die Kommandozeile mit `curl` oder `wget`
  - `curl -k 'https://localhost:8080/json/apartment/getName'`
  - `wget -qO - --no-check-certificate 'https://localhost:8080/json/apartment/getName'`
  - Möglicherweise muss noch
- Im Browser API URLs aufrufen (z.B. Firefox mit JSONovich oder Firebug Plugin)

Das JSON API

Mit dem JSON API "spielen"

- Über die Kommandozeile mit `curl` oder `wget`
  - `curl -k 'https://localhost:8080/json/apartment/getName'`
  - `wget -qO - --no-check-certificate 'https://localhost:8080/json/apartment/getName'`
  - Möglicherweise muss noch
- Im Browser API URLs aufrufen (z.B. Firefox mit JSONovich oder Firebug Plugin)

Beispiel

Lesen des aktuellen Gesamtverbrauch.

Beispiel

HTML Rahmen für die Applikation

```
<?xml version="1.0"?>
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head><title>dSS Metering Demo: Simple</title></head>
<body>
    <h1>dSS Metering Demo: Simple</h1>
    <div id="data">
        Power: <span class="power-info">0 W</span>
    </div>

    <script
        type="text/javascript"
        src="/js/extjs/adapter/ext/ext-base-debug.js">
    </script>
    <script type="text/javascript"
            src="/js/extjs/ext-all-debug.js"></script>
    <script type="text/javascript"
            src="/js/demo.js"></script>
</body>
</html>
```

Beispiel

Verwendung von ExtJS ist naheliegend, da beim dSS schon dabei

Aktueller Verbrauch des Apartment lesen

```
Ext.onReady(function() {
    Ext.Ajax.request({
        url: '/json/apartment/getConsumption',
        success: function(response) {
            console.debug("Received JSON:", response.
                responseText);
        }
    });
});
```

Beispiel

Aktueller Verbrauch des Apartment lesen

```
Ext.onReady(function() {
    Ext.Ajax.request({
        url: '/json/apartment/getConsumption',
        success: function(response) {
            console.debug("Received JSON:", response.
                responseText);
        }
    });
});
```

Beispiel

Aktueller Verbrauch im HTML DOM Baum aktualisieren

```
function onDataReceived(response) {
    var data = Ext.decode(response.responseText),
        consumption = (data &&
                       data.result &&
                       data.result.consumption) || 0,
        el = Ext.get('data').select('.power-info').first();
    el.update("" + Math.round(consumption) + " W");
}

Ext.onReady(function() {
    Ext.Ajax.request({
        url: '/json/apartment/getConsumption',
        success: onDataReceived
    });
});
```

Beispiel

Aktueller Verbrauch regelmässig aktualisieren

```
function onDataReceived(response) {
    var d = Ext.decode(response.responseText),
        consumption = (data &&
                       data.result &&
                       data.result.consumption) || 0,
        el = Ext.get('data').select('.power-info').first();
    el.update("" + Math.round(consumption) + " W");
}

Ext.onReady(function() {
    Ext.TaskMgr.start({
        interval: 3000,
        run: function() {
            Ext.Ajax.request({
                url: '/json/apartment/getConsumption',
                success: onDataReceived
            });
        }
    });
});
```

Umgebung

Umgebung zum testen

- dSS mit echter Hardware
- oder dSS im Simulationsmodus

Umgebung

Wohin mit den Dateien?

- Dateien nach `data/webroot/` kopieren
- Danach sind sie unter `https://localhost:8080/` erreichbar

Dokumentation

Notwendige Dokumentation und Hilfe

- Einstiegspunkt Dokumentationsübersicht im Wiki [digc]
- JSON API Dokumentation, entweder Online [digb] oder dSS Build Verzeichnis (`make doc` gefolgt von `xsltproc doc/json_api.xslt build/doc/json_api.xml > build/doc/json_api.html`)
- Wiki Seite zur Simulation [digd]
- dss-developer Mailingliste [diga]

Weitere Ideen

- Entdecken Sie den Property Tree
- Laden Sie sich Zeitreihen von der Metering Location und erstellen Sie sich daraus ein Diagramm
- Programmieren Sie sich ein Desktop Widget (Dashboard, Sidebar, etc)
- Lösen Sie Events aus

Fazit

Das JSON API ist einfach zu verstehen und zu benutzen.

Fazit

Das API ist gut dokumentiert [digc].

References I

digitalSTROM.org.
dss developer mailinglist.
http://forum.digitalstrom.org/cgi-bin/mailman/listinfo/dss-developer.

digitalSTROM.org.
dss json api dokumentation.
http://developer.digitalstrom.org/releases/dss-0.8.0-doc/dss-0.8.0-json_api.html.

digitalSTROM.org.
dss wiki dokumentationsübersicht.
http://developer.digitalstrom.org/redmine/projects/dss/wiki/Documentation.

References II


digitalSTROM.org.
dss wiki seite zur simulation.
http://developer.digitalstrom.org/redmine/projects/dss/wiki/Simulation.