

How To Use CAPH Data Acceleration API

Published 2014-10-28 | (Compatible with SDK 5.0,5.1 and 2013,2014 models)

Data Acceleration is provided by CAPH Data Acceleration API. The function of Data Acceleration library is high level API for response (loading) time optimization.

Contents

[Prerequisites](#)

[Environment](#)

[Source File](#)

[DAL APIs Overview](#)

[How to Develop](#)

[LocalStorageCache sample](#)
[preloader sample](#)

[Differences between LocalStorageCache and PreLoader](#)

[Open Issues](#)

[Related Documents](#)

Data Acceleration is provided by CAPH Data Acceleration API. The function of Data Acceleration library is high level API for response (loading) time optimization. DAL includes cache and loader. In case of Cache, there are two type of cache, once some resources are inserted into cache, widget can get the resources from the DAL's cache instead of web. The cache divides into two types such as Local Storage and File system. Local Storage cache is a HTML5 local storage cache. File system cache is implemented to use local file system. In case of Loader, there are three type of loader, once loader is on, the loader read some resources from web before widget access resource. So, loading time will be reduced. The loader divides into three types such as Pre loader, Tag loader, and XHR loader. Pre loader pre-loads some resources from the web. Tag loader loads some resources by Tag. XHR loader loads some resources by XHR. The goal of DAL is optimization of response (loading) time. Today, there are so many contents in web site or DTV widget. Also image resources' size is increased in order to display high resolution. For example, the total size of image resources is 3MB. If a widget uses DAL, it can be reduced loading time for resources. But, if a widget doesn't use DAL, it should load and caching image resources at accessing time from web or some repository. Also this widget decline loading performance. DAL help enhance performance response (loading) time aspect.

Prerequisites

To create applications that run on a TV screen, you need: Samsung TV connected to the Internet, SDK or a text editor for creating HTML, JavaScript and CSS files (using Samsung Smart TV SDK is recommended).

Environment

Use Samsung Smart TV SDK to create the application. You can also use the emulator provided with the SDK to debug and test the application before uploading it in your TV. Later, you can run the application on a TV. Note that applications may perform better on the TV than on the emulator.

You can find general instructions for creating applications in [Implementing Your Application Code](#).

Put `caph-level1-unified.js` into your HTML file:

\$CAPH/1.0.0/ caph-level1-unified.js

Put **caph.css** into your HTML file:

\$CAPH/1.0.0/caph.css

Then you may include your application .js file. Let's say your application's JavaScript file is "localstoragecache.js." and it has an app object and app.run() method. Then, you may run your Samsung Smart TV application by putting the following code in the <head> and <body> sections of your HTML, just as follows:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
  <style>
    .boxcls {
      background: green;
    }
  </style>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title>LocalStorageCache samples</title>

    <!-- load caph-level1-unified.min.js, caph.css -->
    <script src="$CAPH/1.0.0/caph-level1-unified.min.js" type="text/javascript"></script>
    <link href="$CAPH/1.0.0/base.css" rel="stylesheet" type="text/stylesheet">
    <!-- load the app -->
    <script src="LocalStorageCache.js"></script>
  </head>
  <body onload="app.run();">
  </body>
</html>
```

Source files mentioned above are explained in the following table.

File	Description
caph-level1-unified.min.js	It is include all kinds of is basic widget of UI will be used. It includes.
LocalStoragecache.js	The localstoragecache application source file.
caph.css	The class style of caph.wui.

Source File

Source files in directory are explained in the table.

File	Description
caph-level1-unified.min.js	Basic component of UI. It includes external libraries also.
localstoragecache.js preloader.js	The app source file

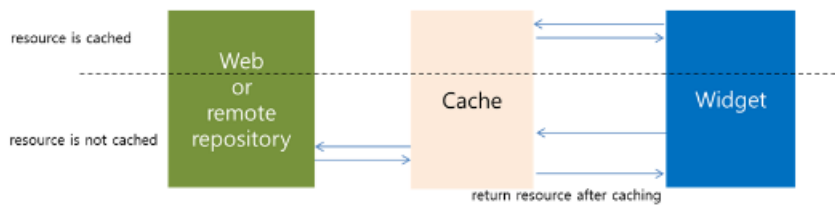
DAL APIs Overview

Local storage cache:

Provides caching for some resource files. It uses HTML5 local storage cache. Once some resources are inserted into local storage cache, widget that uses local storage cache gets the resources from local storage cache instead of web.

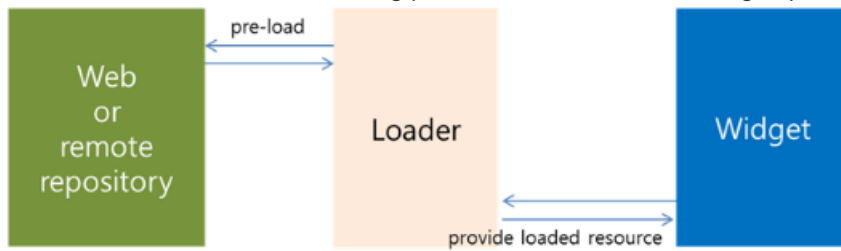
File system cache:

Provides caching for some resource files. It uses local file system. Once some resources are inserted into file system cache, widget that uses file system cache gets the resources from file system cache instead of web.



Pre loader:

Pre-loads some resources. Loading procedure is divided according to priority, dependency or no condition loading.



Tag loader:

This is loads some resources by tag.

XHR loader:

This is loads some resources by XHR.

How to Develop LocalStorageCache sample

You should import the files to *.html file.

```

1  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
2  <html>
3  <style>
4    .boxcls {
5      background: green;
6    }
7  </style>
8  <head>
9    <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
10   <title>LocalStorageCache Samples</title>
11
12   <!-- load caph-level1-unified.min.js, caph.css -->
13   <script src="$CAPH/1.0.0/caph-level1-unified.min.js" type="text/javascript"></script>
14   <link href="$CAPH/1.0.0/base.css" rel="stylesheet" type="text/css">
15
16   <!-- load the app -->
17   <script src="LocalStorageCache.js"></script>
18 </head>
19 <body onload="app.run();">
20 </body>
21 </html>

```

The codes describe as follow.

Line 13– 14	Load caph-level1-unified.min.js, caph.css.
Line 16	Load app source file.

The LocalStorageCache.js file is as below:

```

1  /**
2   * DAL Localstoragecache sample
3   */
4   var app = app || {};
5
6   app.run = function() {
7       var Localstoragecache = caph.core.dal.LocalStorageCache;
8       var localstoragecache = new Localstoragecache();
9
10      var blobResponseArray = [];
11      var imgURLArray = [];
12      imgURLArray.push('http://memory.loc.gov/ammem/collections/habs_haer/images/eal41.jpg');
13      imgURLArray.push('http://memory.loc.gov/ammem/collections/habs_haer/images/baz132.jpg');
14
15      function saveBlob(url) {
16          var xhr = new XMLHttpRequest();
17          xhr.onreadystatechange = function() {
18              }
19          xhr.responseType = 'blob';
20          xhr.open(GET,url,true);
21          xhr.send();
22      };
23
24
25      localstoragecache.init("localstoragecache", 1024*1024*5/2, function() {
26          for (var i = 0; i < imgURLArray.length; i++) {
27              saveBlob(imgURLArray[i]);
28          }
29      });
30
31      //blobResponseArray[i] contains Blob for each bitmap image resources in URL, imgURLArray[i]
32      for (var i = 0; i < blobResponseArray.length; i++) {
33          blobOption.url = imgURLArray[i];    blobOption.blob = blobResponseArray[i];
34          blobOption.size = blobResponseArray[i].size;
35
36          localstoragecache.insertBlob(blobOption);
37      }
38
39      for (var i = 0; i < imgURLArray.length; i++) {
40          var br = localstoragecache.find(imgURLArray[i]);
41          if (br){
42              var br = localstoragecache.get(imgURLArray[i]);
43              if (br){
44                  var url = item;
45                  var img = document.createElement("localstoragecache");
46                  img.url = url;
47
48                  localstoragecache.insertBlob(blobOption);
49              }
50          }
51      }
52  };

```

The codes describe as follow.

Line 7	Import the components will be used.
--------	-------------------------------------

Line 8	New the LocalStorageCache object.
Line 10-14	Define array and insert default url value.
Line 16-20	Init the LocalStorageCache object.
Line 23-29	Do the insertBlob operation of cache, you can also do other operation about cache such as find, get and remove.

preloader sample

Line 7	Import the components that will be used.
Line 8	New the preloader object.
Lines 10-13	Define array and insert default url value.
Line 15-26	Make blob type data based on image.
Line 28	Init the preloader object.
Lines 29-38	Load resources and start preloader.
Lines 40-42	Complete callback function. If subscribe function is completed, invoke this callback.
Lines 43-45	Error callback function. If subscribe function occur error, invoke this callback.

You should import the files to *.html file.

```

1  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
2  <html>
3  <style>
4      .boxcls {
5          background: green;
6      }
7  </style>
8  <head>
9      <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
10     <title>preloader samples</title>
11
12     <!-- load caph-level1-unified.min.js, caph.css -->
13     <script src="$CAPH/1.0.0/caph-level1-unified.min.js" type="text/javascript"></script>
14     <link href="$CAPH/1.0.0/base.css" rel="stylesheet" type="text/stylesheet">
15
16     <!-- load the app -->
17     <script src="preloader.js"></script>
18 </head>
19 <body onload="app.run();">
20 </body>
21 </html>

```

The codes describe as follow.

Lines 12–16	Import the dependencies files.
Line 18	Load app source file.

The preloader.js file is as below:

```


```

```

1  /**
2   * DAL preloader sample
3   */
4   var app = app || {};
5
6   app.run = function() {
7       var Preloader = caph.dal.Preloader;
8       var preloader = new Preloader ();
9
10      var blobResponseArray = [];
11      var imgURLArray = [];
12      imgURLArray.push('http://memory.loc.gov/ammem/collections/habs_haer/images/eal41.jpg');
13      imgURLArray.push('http://memory.loc.gov/ammem/collections/habs_haer/images/baz132.jpg');
14
15      function saveBlob(url) {
16          var xhr = new XMLHttpRequest();
17          xhr.onreadystatechange = function() {
18              if (xhr.readyState == 4 && xhr.status == 4){
19                  var blob = xhr.response;
20                  blobResponseArray.push(blob);
21              }
22          }
23          xhr.responseType = 'blob';
24          xhr.open(GET,url,true);
25          xhr.send();
26      }
27
28      preloader.init();
29      preloader.setFlagThrottled(true);
30      preloader.setMaxIssuePerTurn(1);
31      preloader.setFlagSync(true);
32      preloader.subscribe('oncomplete', onComplete);
33      preloader.subscribe('onerror', onError);
34      rc = preloader.loadWithPriorityManifest({
35          files : imgURLArray
36      });
37
38      preloader.start();
39  };
40  function onComplete() {
41  }
42  }
43  function onError() {
44  }
45  }

```

The codes describe as follow.

Line 7	Import the components that will be used.
Line 8	New the preloader object.
Lines 10-13	Define array and insert default url value.
Line 15-26	Make blob type data based on image.

Line 28	Init the preloader object.
Lines 29-38	Load resources and start preloader.
Lines 40-42	Complete callback function. If subscribe function is completed, invoke this callback.
Lines 43-45	Error callback function. If subscribe function occur error, invoke this callback.

Differences between LocalStorageCache and PreLoader

localStoragecache test

insert

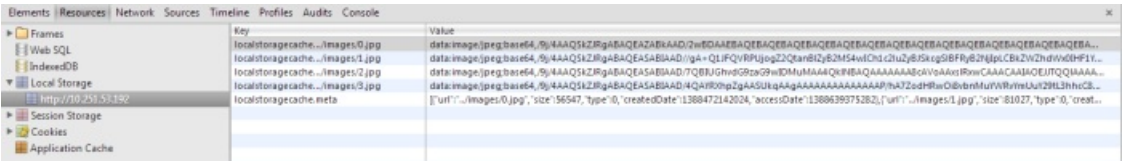
read

preloader test

preload

read

LocalStorageCache and PreLoader have differences concept aspect. LocalStorageCache uses HTML5 local storage cache, so it saves contents such as image, text, binary, and so on. PreLoader can take contents on the Web or other remote repository before using it. The basic difference point is the timing of take the contents. LocalStorageCache is for enhancement of loading time that the contents are already loaded. Below are inserted items (Google Chrome's inspector). If click "insert" button, then the items are inserted HTML5 localstoragecache.



But, PreLoader is for enhancement of loading time that the contents are not loaded. It is useful if pre-loading another web page's contents and web page will be visit soon. Because don't have to take contents at visit web page time, loading time will be shorten.

Open Issues

None

Related Documents

- How to Use CAPH WUI Widgets
- How To Customize CAPH WUI Widgets
- How To Use CAPH WUI Widgets With HTML
- How To Use Animation With CAPH WUI Widgets
- How To Integrate CAPH WUI Widgets With jQuery Library