# HP Propel

Software version 2.00

# Customizing Launchpad

## Contents

Document Release Date: July 2015

Software Release Date: July 2015

## Introduction

Your administrator can customize the display of the HP Propel Launchpad to meet your organization's needs.

**Launchpad Configuration File**
One of the primary configuration files involved in Launchpad customization is the configuration file: `/opt/hp/propel/launchpad/conf/dashboard.json`. A detailed example can be found in the **Launchpad Customization** section.

**Note:** The HP Propel Launchpad service must be restarted after changes are made to the dashboard.json file. To do this, enter `service launchpad restart` from the command prompt. Users must log out, clear their browser cache, and log in again to see the changes.

**Glyph Icons**
The Portal styling framework uses a collection of open source glyph icons that can be used throughout the application. See **Appendix A: Glyph Icons**.

## Corporate or Organization Logo

A logo or picture for your company or organization can be used in the HP Propel Launchpad and Portal. It will be displayed on the HP Propel login page, in Launchpad, in the upper left-hand corner of Portal views, and in administrator UIs where appropriate.

Add a logo or picture as follows:
- Access the Organization administration UI. (Select **Identity** from Launchpad.)
- Create a new organization or select an existing organization to modify.
- Select **General Information**.
- Enter the URL for a picture that represents the organization in the **Organization Picture URL** box.
- Click **Save**.

**Note:** The HP Corporate logo is not part of the out-of-the-box content. It is a custom image.

## Launchpad Customization

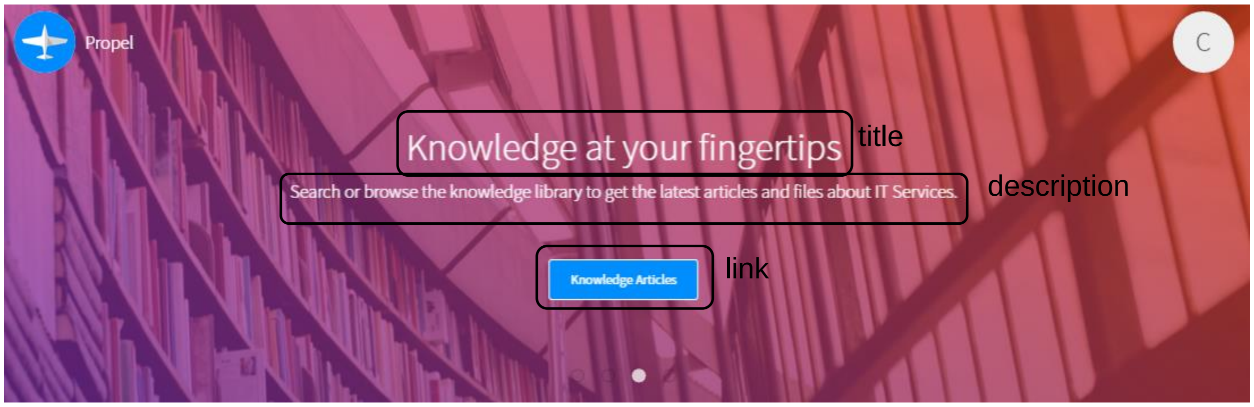
### Banner

Launchpad banner customization is achieved by modifying the configuration file typically located at `/opt/hp/propel/launchpad/conf/dashboard.json`. A version of the `dashboard.json` file can optionally be customized for each organization. The filename should follow naming convention `dashboard.<organization name>.json`. For example, an organization named ACME would have a customized file named `dashboard.ACME.json`. Launchpad will automatically look for this file when a user from organization ACME logs in. If this file does not exist, the contents of `dashboard.json` will be used.

**Note:** The HP Propel Launchpad service must be restarted after changes are made to the dashboard.json file.

As part of Launchpad banner customization, multiple banners can be defined. Each will be displayed for the specified time interval on a rotating basis. Customization is done by modifying the **header** content in the dashboard.json configuration file typically located at `/opt/hp/propel/launchpad/conf/dashboard.json`.

Following is an example banner with properties labeled for easy identification.



Knowledge at your fingertips

Search or browse the knowledge library to get the latest articles and files about IT Services.

Banner **header** configurations file properties:

**Note:** Configuration properties are typically optional. At least one entry in **items** must be defined for the banner to be useful. Values will be empty if **default** properties are not defined and individual **items** properties are also not defined.

| Value Items | Description |
|---|---|
| default | Specifies the default value for the banner properties. Any **items** properties not specified will use these default values. Default: not specified. |
| Interval | If multiple banner **items** are defined, this value is the time in milliseconds to display one banner before switching to the next. Default: 5000 milliseconds. |
| items | Array or list of banners. Default: banner not displayed. |
| id | User defined string to identify this configuration file item. |
| title | Title to be displayed on the banner. Default: “No data to show”. |
| role | User’s role. Can be: CONSUMER, SUPPORT, CONTENT_ADMIN, CATALOG_ADMIN, AGGREGATION_ADMIN, SUPER_IDM_ADMIN |
| description | Description to be displayed under the title. Default: no description displayed. |
| background | Specify one or more className values and optional image URL. Default: gray background. |
| className | <See list of valid values following this table.> |
| url | URL reference to an image. Can be an internal or external link. |
| link | URL link, or a link from current page to another. |
| url | URL address of next page. Can be an internal or external link. |
| target | Optional. Web page or browser target for the URL link. **null** or undefined default to using the same page (“_self”). Possible values are: • _blank: Opens the link in a new browser window or tab. • _self: Opens the link in the current browser frame. • _parent: Opens the link in the browser parent frame. • _top: Opens the link in the full browser window. • <framename>: Opens the link in the named browser frame. |

| | |
|---|---|
| label | Text label displayed on the link. Can be a static text string or a localized string predefined in the application resource bundle. |
| template | Template to use for the banner, either **default** or **halfColumn**. **default** uses all information defined for the banner. **halfColumn** displays content in the left half of the banner space, thereby allowing more of the background content to be viewed. |

**Banner className values**

The following predefined **className** values are available:

- Text colors:
  - **text-white**
  - **text-black**
  - **text-green**
  - **text-dark-blue**
  - **text-light-blue**
  - **text-lemon**
  - **text-rock**
  - **text-orange**
  - **text-red**
  - **text-apple-green**
  - **text-pink**
  - **text-light-red**

- Background colors:
  - **green**
  - **dark-blue**
  - **light-blue**
  - **lemon**
  - **orange**
  - **rock**
  - **white**
  - **black**

- Background images:
  - **bg-blur**
  - **bg-red-sky**
  - **bg-dots**
  - **bg-success**
  - **bg-warning**
  - **bg-critical**
  - **bg-1** through **bg-21** – collection of background images

- Other background styling:
  - **background-stretch** – Stretch background image as needed to fill banner background. Requires that a background image be specified.
  - **background-repeat** – Repeat the background image horizontally and vertically to fill the area. Requires that a background image be specified.
  - **background-cover** – Scale the background image to cover the background. Note that some of the image might not be visible, depending on the image dimensions.

Example configuration file content format:

```
header: {
  interval: Number, //in milliseconds
  items: [
    {
      "id": String,
      "title": String,
      "role": String,
      "description": String,
      "background": {
        "className": String,
        "url": String
      }
      "link": {
        "label": String,
        "url": String
      },
    },
    ...
  ]
}
```

Following is example content that could be included in the configuration file found at `/opt/hp/propel/launchpad/conf/dashboard.json`. Each banner displays for 5 seconds before switching to the next banner.

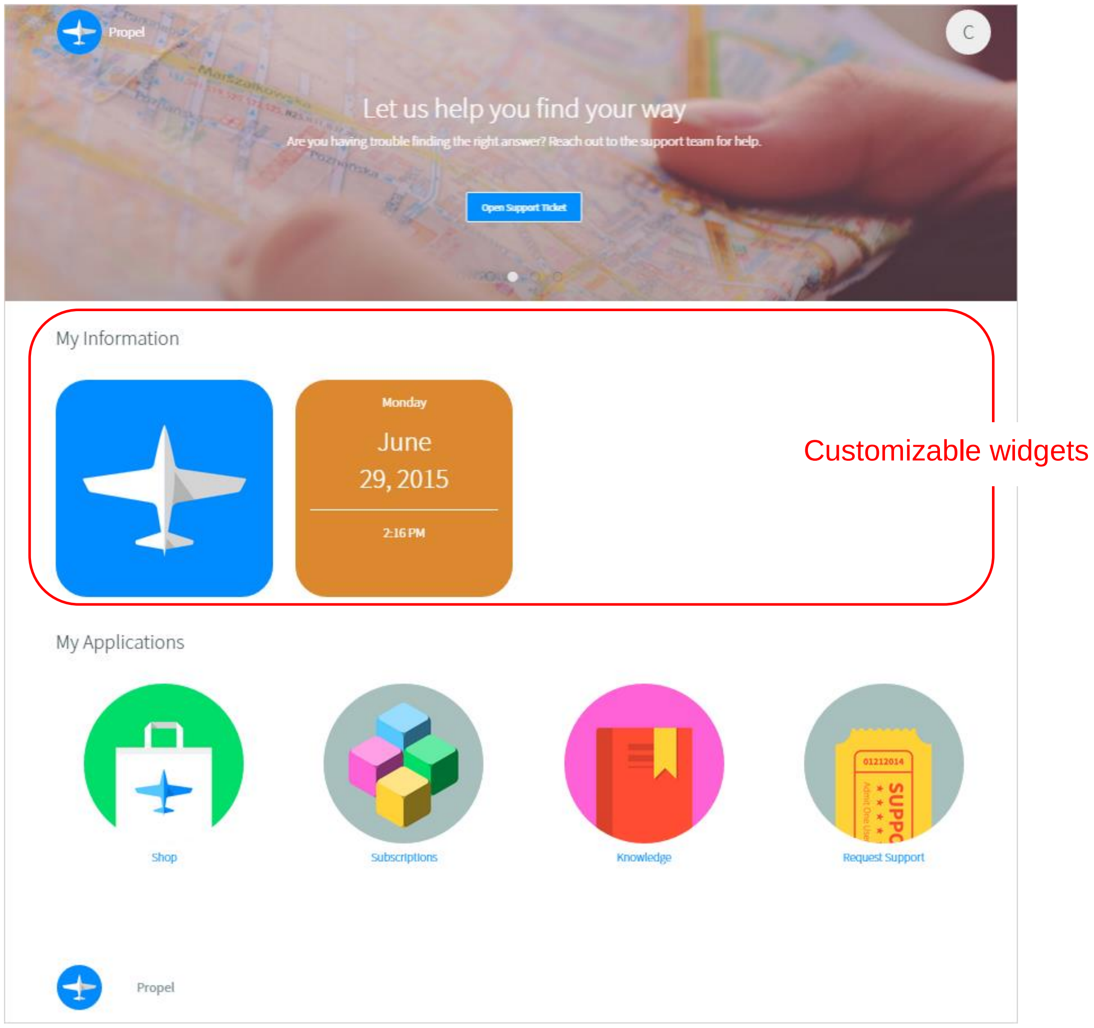
```
{
  "header": {
    "interval": 5000, // 5 seconds - speed of Carousel autoplay
    "items": [
      {  //The first banner uses assets available in the Portal, and uses the
default template.
        "id": "dashboard-banner_service",
        "title": "lpd.dashboard.banner.service.title",
        "description": "lpd.dashboard.banner.service.description",
        "role": ["CONSUMER", "SUPPORT"],
        "background": {
          "url": "assets/launchpad/images/banner-01.png"
        }
      },
      { //The second banner specifies the halfColumn template, and includes a
URL link button.
        "id": "dashboard-banner_support",
        "title": "lpd.dashboard.banner.support.title",
        "description": "lpd.dashboard.banner.support.description",
        "role": ["CONSUMER", "SUPPORT"],
        "template": "halfColumn",
        "link": {
          "label": "lpd.dashboard.banner.support.link",
          "url": "https://localhost:8089/#/support",
          "className": "btn-md bg-primary"
        },
        "background": {
          "className": "text-black background-cover",
          "url": "assets/launchpad/images/banner-02.png"
        }

      }
```

```
        ]
    }
}
```

## Widgets

Let’s look now at customizing widgets in Launchpad.

Your administrator can add custom widgets to the **My Information** section of Launchpad, such as the HP Propel link and calendar widgets shown in the following image. Widgets can only be added to Launchpad; they cannot be added to applications or components, such as the Shop app.

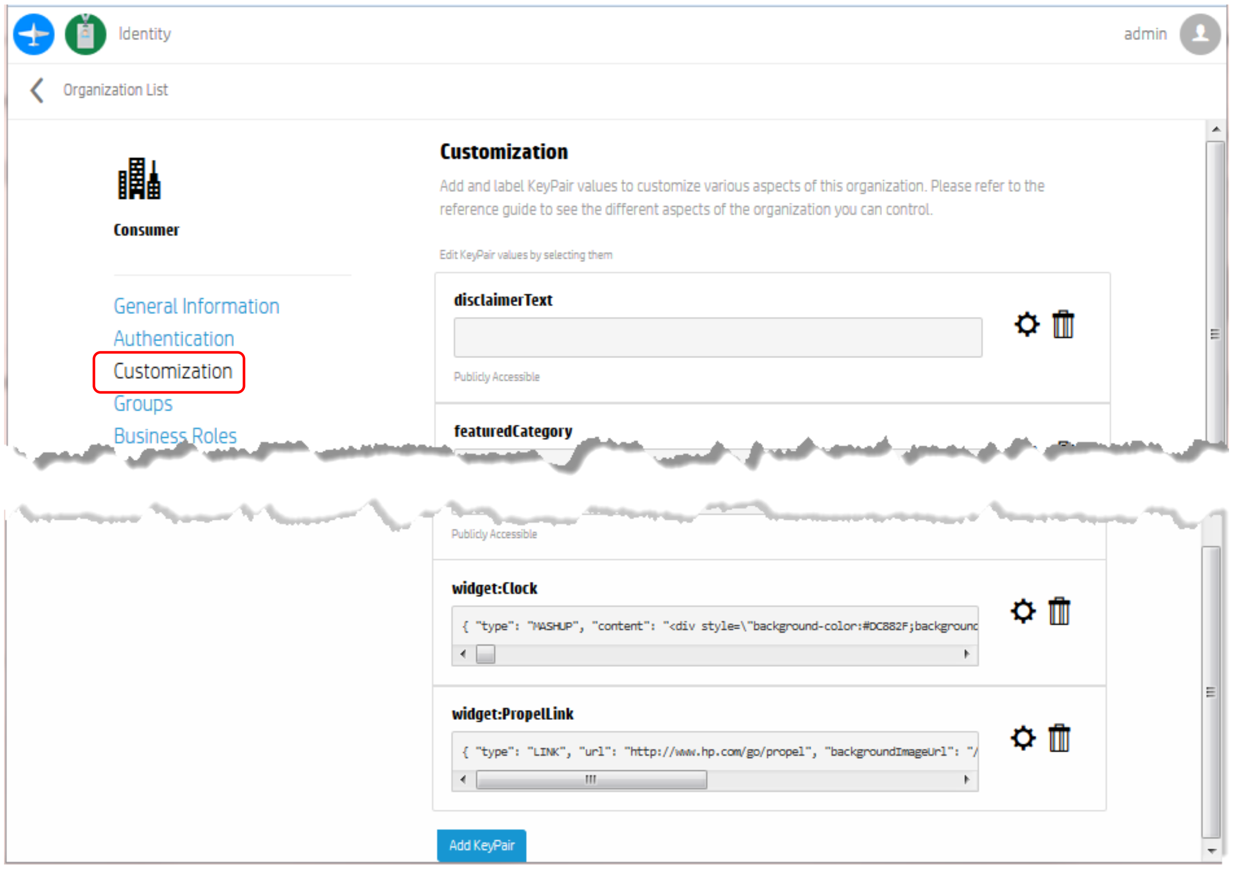


HP Propel Launchpad supports the following types of custom widgets:

- Link Widget
- Mashup Widget

The size of a widget in Launchpad is fixed. Note that it is recommended that you not change the size of even complex mashup widgets as unexpected behavior can result.

Custom widgets are defined per organization under **Customization** in the **Identity** (Organizations) area of HP Propel as shown in the following, and will be available to all users in that organization. Click on the **Add KeyPair** button at the bottom of the Customization screen to create new widgets. See the *HP Propel Organizations Help* for more information.



Customization

Add and label KeyPair values to customize various aspects of this organization. Please refer to the reference guide to see the different aspects of the organization you can control.

Edit KeyPair values by selecting them

disclaimerText

Publicly Accessible

featuredCategory

Publicly Accessible

widget:Clock

```
{ "type": "MASHUP", "content": "<div style=\"background-color:#DC882F;backgrounc
```

widget:PropelLink

```
{ "type": "LINK", "url": "http://www.hp.com/go/propel", "backgroundImageUrl": "/
```

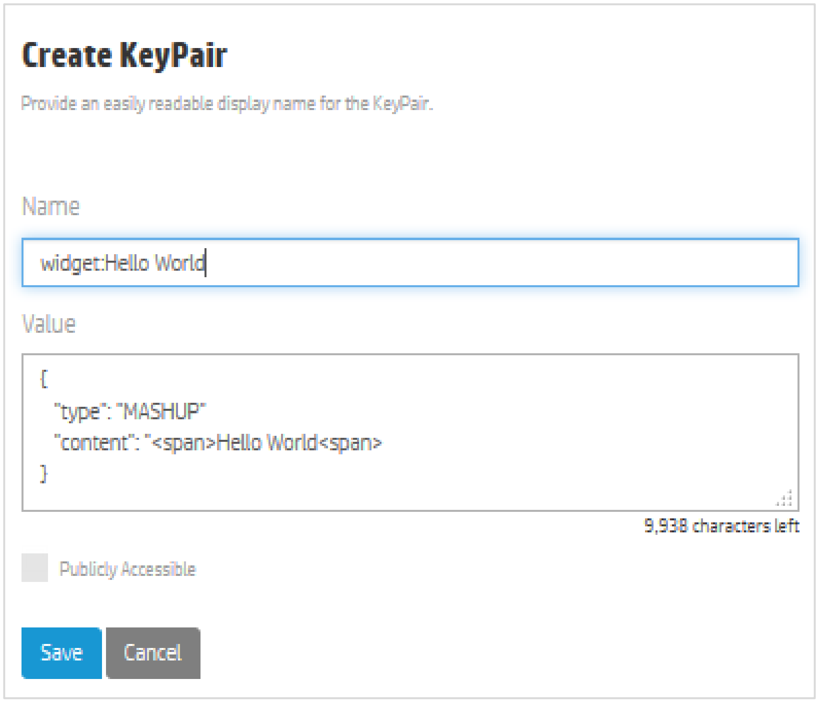
Note that there are entries for two custom widgets corresponding to the two custom widgets in the previous Launchpad image.

## Creating Widgets

All widget types are created in HP Propel as key:value pairs in the Customization section of the Organization List view.

The key, or name, provided must be prefixed with **widget:** in order to identify this key pair as a widget. The value of the key pair will always be a JSON object and specific values required will depend on the widget type as detailed in following sections.

Figure 1: Create KeyPair Example



Create KeyPair

Provide an easily readable display name for the KeyPair.

Name: widget:Hello World

Value:

```
{
  "type": "MASHUP"
  "content": "<span>Hello World<span>
}
```

9,938 characters left

**Link Widget**

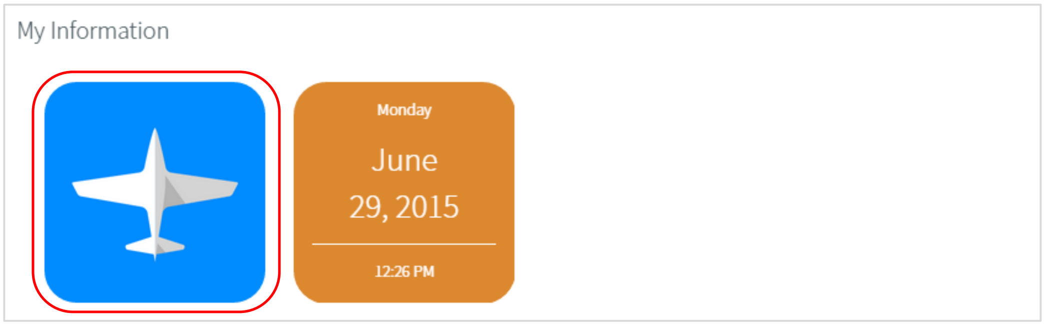
A link widget is a quick-and-easy shortcut to access a URL from Launchpad. As with all custom widgets, link widgets display in the **My Information** row of Launchpad.

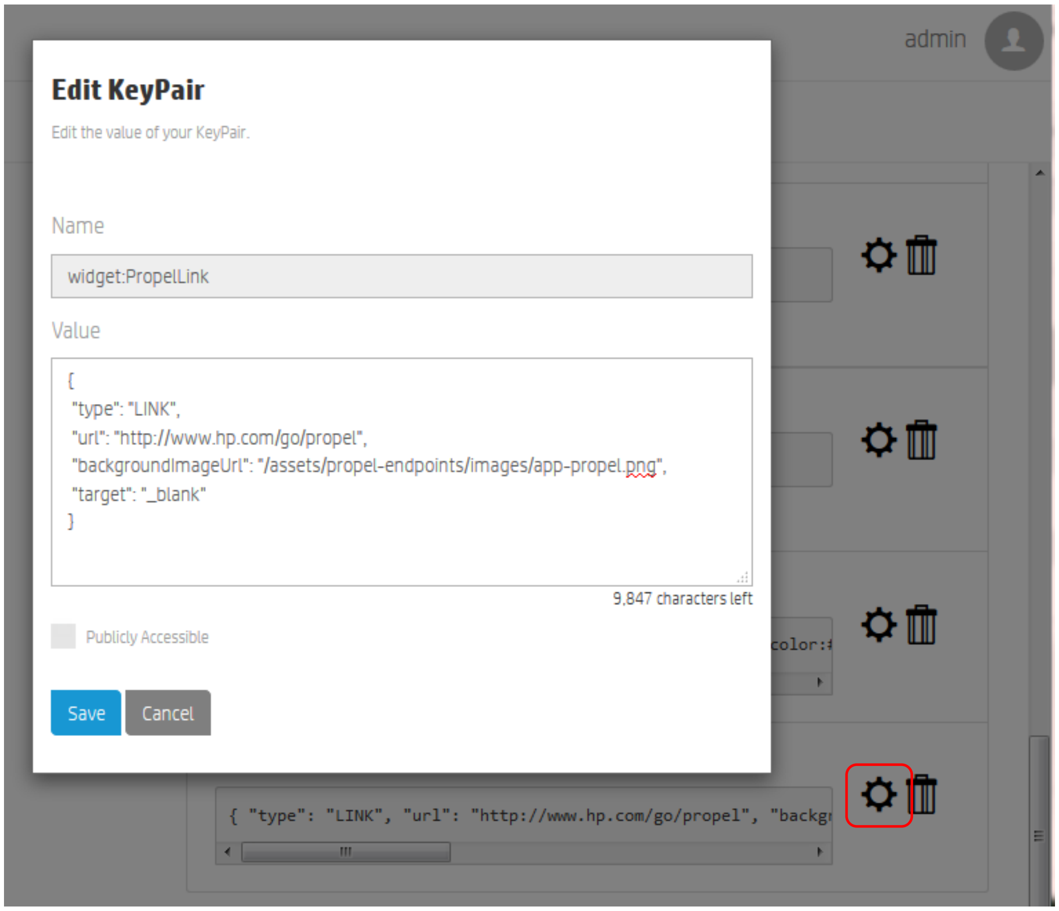
Use the **Create KeyPair** dialog to create a custom link widget. The **type** value must be set to **LINK**. With link widgets you customize the link URL, display name, icon image, background image, and the HTML target.

| Value Items | Description |
|---|---|
| type | Must be set to **LINK** |
| url | URL that the link references in Launchpad. |
| displayName | Optional name to be displayed on the widget. |
| iconUrl | Optional URL of an icon that displays near the center of the widget. |
| backgroundImageUrl | Optional URL of an image that fills the background of the widget. |
| target | Optional target attribute of the <link> element that appears in Launchpad and that controls the browser window in which the link will open. Valid values for the target attribute are defined in the HTML specification. |

**Tip:** If new to creating widgets, review existing custom widgets in Launchpad and associated details in the **Edit KeyPair** dialog to help in creating new widgets.

Following is the content for the custom HP Propel link widget in Launchpad.





**Edit KeyPair**

Edit the value of your KeyPair.

Name

widget:PropelLink

Value

```
{
"type": "LINK",
"url": "http://www.hp.com/go/propel",
"backgroundImageUrl": "/assets/propel-endpoints/images/app-propel.png",
"target": "_blank"
}
```

9,847 characters left

Publicly Accessible

Save Cancel

```
{ "type": "LINK", "url": "http://www.hp.com/go/propel", "backgr
```

### Mashup Widget

Mashup widgets give administrators the ability to define more complex widgets to meet an organization’s needs. These might be generally useful widgets such a calendar or clock, or perhaps widgets for HP Propel-specific information such as recent subscriptions.

Mashups widgets require **type** be set to **MASHUP**. Mashup widgets contain one other key:value pair, **content**, which contains the source for your mashup.

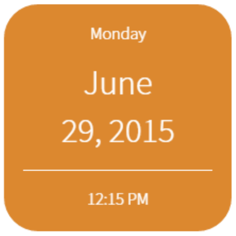
**Note:** All double quotation marks (“) must be escaped (\”) in the mashup widget content.

| Value Items | Description |
|---|---|
| type | Must be set to **MASHUP**. |

| | |
|---|---|
| content | The HTML and JavaScript code for the mashup.<br>When using iFRAME in a mashup widget, note the following:<br>• iFrames that serve HTML pages that have the same URL structure as Launchpad will work properly. The same URL structure means that the pages are placed in the following directory:<br>`/opt/hp/propel/launchpad/dist`<br>For example, to correlate to the following URL structure:<br>`https://server:8089/widgets/sample/index.html`<br>You would place your pages in the following location:<br>`/opt/hp/propel/launchpad/dist/widgets/sample/index.html`<br>• iFrames that serve external NON-HTTPS content will be blocked by the browser. The specific error will vary based on client browser security.<br>• iFrames that serve external HTTPS content that contains mixed HTTP and NONHTTPS content will be blocked by the browser. The specific error will vary based on client browser security.<br>• iFrames that serve external HTTPS content will work only if the following are true:<br>– The remote site must not specify x-frame-options DENY in the response header.<br>– If the content is not of the same origin domain, and the remote site has not specified x-frame-options SAMEORIGIN, the content will display properly. |

Example custom widget: Clock

```
{

    "type": "MASHUP",

    "content": "<div style=\"background-color:#DC882F;background-image:
url(''); color: #FFFFFF; font-size: 18px;background-repeat: no-repeat;
padding-left: 20px;padding-top: 20px;padding-right: 20px;padding-bottom:
20px;line-height: .7;height:100%;\"><p id=\"weekday\" style=\"font-size:
18px;\">-</p><h1 id=\"month\" style=\"font-size: 36px;color:white;\">-
</h1><h1 style=\"font-size: 36px;color:white;\"><span id=\"day\">-</span>,
<span id=\"year\">-</span></h1><hr style=\"color: #FFF;border-
color:white;\"/><p style=\"font-size: 18px;\"><span id=\"hmm\">-</span> <span
id=\"ampm\">-</span></p></div><script type=\"text/javascript\">function
startClock() { var months = [ \"January\", \"February\", \"March\",
\"April\", \"May\", \"June\", \"July\", \"August\", \"September\",
\"October\", \"November\", \"December\" ]; var weekdays = [\"Sunday\",
\"Monday\", \"Tuesday\", \"Wednesday\",
\"Thursday\",\"Friday\",\"Saturday\"]; function updateClock() { var now = new
Date(); var hours = now.getHours(); var minutes = now.getMinutes(); var ampm
= \"AM\"; if (Math.floor(hours/12)==1) { ampm = \"PM\"; hours = hours - 12; }
if (hours == 0) { hours = 12; } var ms = \"\" + minutes; if (ms.length == 1)
{ ms = \"0\" + ms; } try { document.getElementById(\"weekday\").textContent =
weekdays[now.getDay()]; document.getElementById(\"month\").textContent =
months[now.getMonth()]; document.getElementById(\"day\").textContent =
now.getDate(); document.getElementById(\"year\").textContent =
now.getFullYear(); document.getElementById(\"hmm\").textContent = hours +
\":\" + ms; document.getElementById(\"ampm\").textContent = ampm; } catch
(err) { clearInterval(clockInterval); } } updateClock(); var clockInterval =
setInterval(updateClock, 1000);}startClock();</script>"
}
```



### Limitations and Requirements

From a performance perspective, additional widgets will take more time to retrieve the associated data. This will result in slower rendering of the Launchpad. This impacts actions such as:

- Login time.
- Refreshing the Launchpad view.
- Returning to Launchpad.

When you create custom widgets, be aware of the following limitations and requirements:

- Variables are shared by widgets. If you are using the same variable name, e.g. list, in multiple widgets, unexpected behavior will result if “list” is assigned different values in the widgets’ content.
- Widgets cannot be shared between organizations. Widgets must be created for each organization.

### Additional Ways to Customize Widgets

There are several ways to customize widgets in Launchpad. As long as you adhere to the security restrictions for running external scripts on HTTPS you can implement widgets any way you want. However, some approaches are simpler than others.

For convenience, custom widgets have access to jQuery and SugarJS. When you use the jquery.ajax() function to make calls to the HP Propel Launchpad APIs, the user session authorization token will be automatically included. You do not need to extract the user data from the session.

Optionally, you can extend this code to include an inline <style> tag for more complex CSS cases:

```
</script><style type=\"text/css\">#expensiveSubsDiv>p{color:black}</style>
```

When you append a style tag, wrap your widget in a top level div, such as `<div id=\"expensiveSubsDiv\">`. This will ensure help your styling does not contaminate elements outside of your widget.

Using jQuery, combined with SugarJS, should make developing widgets relatively simple, even when you use data from HP Public APIs, as shown in the example above. However, there are limitations when running external scripts in an HTTPS environment. Do not expect to be able to copy and paste an iframe from a widget-generating website when running in an HTTPS environment. This is not supported at the browser level.

### Using the IFrame Technique

Due to limitations of widgets you might find that trying to store your entire mashup widget in the **content** value of the body can be quite a challenge when having to escape (\) quotation marks in your source, e.g. `"<div id=\"recentSubs\">…"`. Using the Iframe technique will enable you to create much more complicated widgets with much less frustration.

For example:

```
{
   "type": "MASHUP",
   "content": "<iframe src=\"widgets/IFrameSample/index.html\"
height=\"100%\" width=\"100%\"></iframe>"
}
```

This is an example of code that will leverage the IFrame technique in a mashup widget. It's important to note that the height and width of the IFrame should be set to 100% to fill the space available within the widget. You are still required to escape quotation marks within the **content**; however the issue is significantly reduced by moving the bulk of your source code out of the widget's content in the HP Propel **Organization Customization** area, and onto your HP Propel system. The location for your files is: /opt/hp/propel/launchpad/dist.

In the **dist** folder you can create a subfolder for your custom widgets. In the example above this subfolder was named **widgets**. The related files for this one widget were placed into subfolder **IFrameSample**. Add your .html, .css & .js files and reference the index.html file in your IFrame src attribute as shown above.

**Note**: When using the IFrame technique you will not have access to any scripts used by Launchpad (including jQuery, sugarJS & AngularJS), so you will need to reference a public version of any scripts or libraries your widget uses, or more ideally, search the mpp-ui/dist/bower_components folder and reference available libraries there.

**Jquery:** `<script src="/bower_components/jquery/dist/jquery.min.js"></script>`

**SugarJS:** `<script src="/bower_components/sugar/release/sugar.min.js"></script>`

**AngularJS:** `<script src="/bower_components/angular/angular.min.js"></script>`

You will have access to the HP Propel specific APIs as long as your IFrame source code is on the same domain as Launchpad (that is, as long as the source code for the IFrame is in the portal dist folder listed above)

Following is the content of the html, css & js files related to this sample IFrame widget.

**widgets/IFrameSample/index.html:**

```
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" href="style.css" />
  <!—You can optionally use the MPP-Theme styles by uncommenting the
following line -->
  <!-- <link rel="stylesheet" href="/bower_components/mpp-
theme/dist/main.css"> -->
  <script src="/bower_components/jquery/dist/jquery.min.js"></script>
</head>
<body>
  <script src="script.js"></script>
  <div id="recentSubs">
    <b>
    <h5>
      Recent Subscriptions
    </h5>
  </b>
    <ul id="list">
    </ul>
  </div>
</body>
</html>
```

**Widgets/IFrameSample/style.css**

```
body {
  background-color:#7e7e7e;
  padding:10px;
  color:white;
  font-weight:500;
}
h5 {
  margin:0px;
  padding-bottom:10px;
  border-bottom:1px solid white;
  font-weight:500;
}
ul {
  padding-left:0px;
  margin:0px;
}
ul > li {
  display:block;
  padding-top:10px;
  padding-bottom:10px;
  border-bottom:1px solid rgba(255,
  255,
  255,
  0.5);
}
ul > li:last-child {
  border-bottom:none;
}
a {
  color:white;
  text-decoration:none;
}
a:visited {
  color:white;
}
a:hover,
a:focus {
  color:#1897d3;
}
```

### widgets/IFrameSample/script.js:

```
function getSubscriptions(options) {
  $.ajax({
    type: "GET",
    url: "/api/subscription?count=5&offset=0&sort=newest",
    success: function (data, textStatus, jqXHR) {
      if (options.success) {
        options.success(data);
      }
    },
    error: function (jqXHR, textStatus, errorThrown) {
      console.log("Widget Error: Failed to retrieve 5 recent subscriptions");
    }
  });
}

function htmlEncode(value) {
  //create an in-memory div, set its inner text(which jQuery automatically encodes)
  //then grab the encoded contents back out.  The div never exists on the page.
  return $('<div/>').text(value).html();
}

function htmlDecode(value) {
  return $('<div/>').html(value).text();
}

function render() {
  getSubscriptions({
    success: function (data) {
      var $list = $('#list');
      $list.empty();
      if (data && data.length > 0) {
        for (var i = 0; i < data.length; i++) {
          var name = data[i].name;
          var id = data[i].id;
          var url = "/subscription/" + id;
          if (i < data.length - 1) {
            $list.append("<li><a class='ellipses widget-label' target='_parent'
href='" + url + "'><p>" + htmlEncode(data[i].name) + "</p></a></li>");
          } else {
            $list.append("<li><a class='ellipses' target='_parent' href='" + url +
"'>" + htmlEncode(data[i].name) + "</a></li>");
          }
        }
      } else {
        $list.append("<li>No Recent Subscriptions Available</li>");
      }
    }
  });
}
render();
```

## Appendix A: Glyph Icons

The Launchpad styling framework uses a collection of open source glyph icons that can be used throughout the application. To reference these, visit the following open source site for information: **http://fortawesome.github.io/Font-Awesome/**.