Layar Vision Demonstration

This demonstration layer is a simple example implementation of the exciting new features found in Layar Vision. The official documentation covering the entire API is located at: http://layar.pbworks.com

To see this example in action start the Layar app (Layar Vision Android beta client, Layar v6.0 and above) on your device and search for the layer "Cantine" or see it in action in the Layar Vision video on youtube

Requirements:

This demonstration requires a publicly accessible webserver that supports PHP.
Basic knowledge about software development and networking is preferred.

You will also need to have a Layar developer account and be logged into the Layar Publishing Environment.

Getting started

For a layer to actually work the Layar app needs input. In order to get that input it performs a request to the Layar servers and receives the data related to that particular layer. The Layar servers function as a proxy to the server that actually hosts the layer. Please see Layar Platform Architecture Overview for more information.
Based on the parameters specified with the getPOIs request, the layer server will return the data that is relevant to the client.

Step one – preparing your layer server

Let's start by making the layer available on your web space.

Upload the contents of the **PHP directory** within the archive attached to this document to your server. The directory to which you upload should be accessible from the web. The index.php should be called by Layar server.

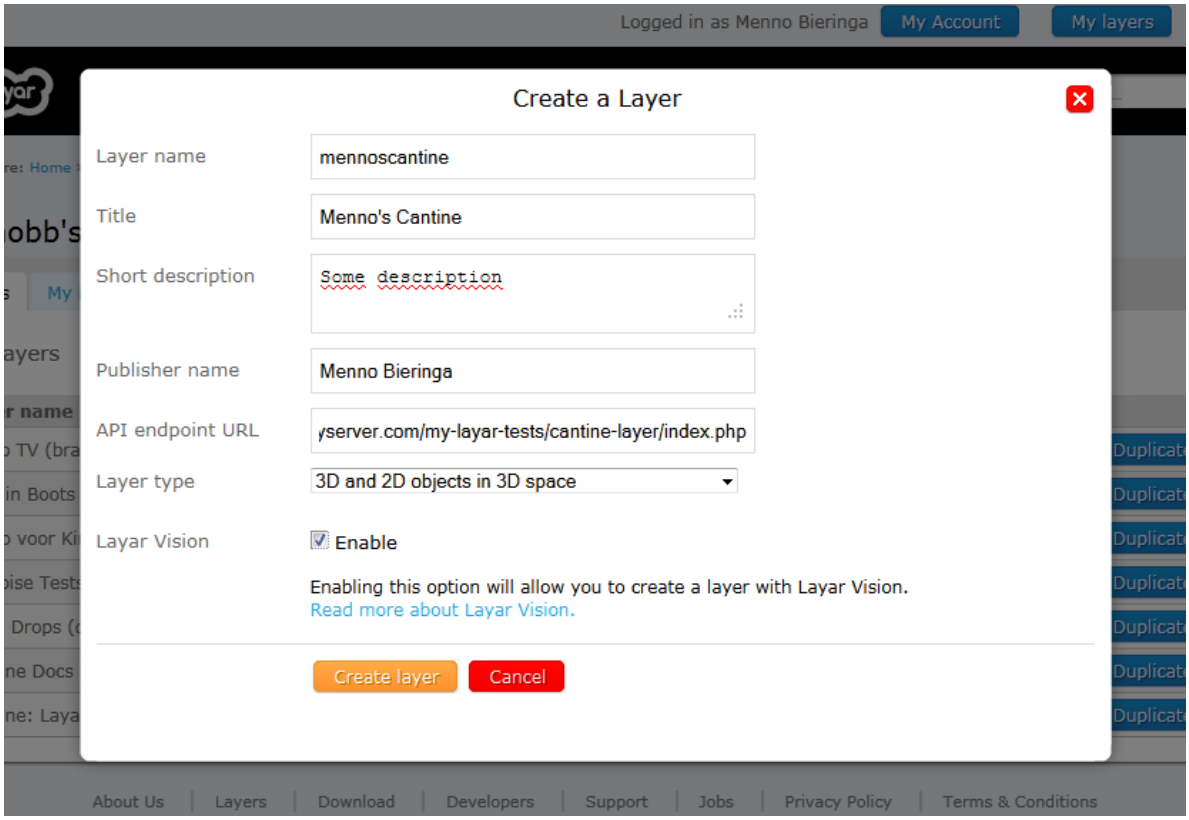
For example: http://myserver.com/my-layar-tests/cantine-layer/index.php

When calling that URL your browser should now display a bunch of JSON, or actually show you the JSON object in a navigable representation.

Step 2 – Creating your layer on the server

The layer server needs to know about your layer so let's define your layer on the server.

Make sure you are logged in to www.layar.com with your developer account, press "Create a new layer" and enter the details of your layer. The only thing you really need to copy from the screenshot below is the Layer Type and Layer Vision values. Also make sure you enter the URL to your webspace (See step 1)



Now click "Create a layer" and in an instant your layer will be created.

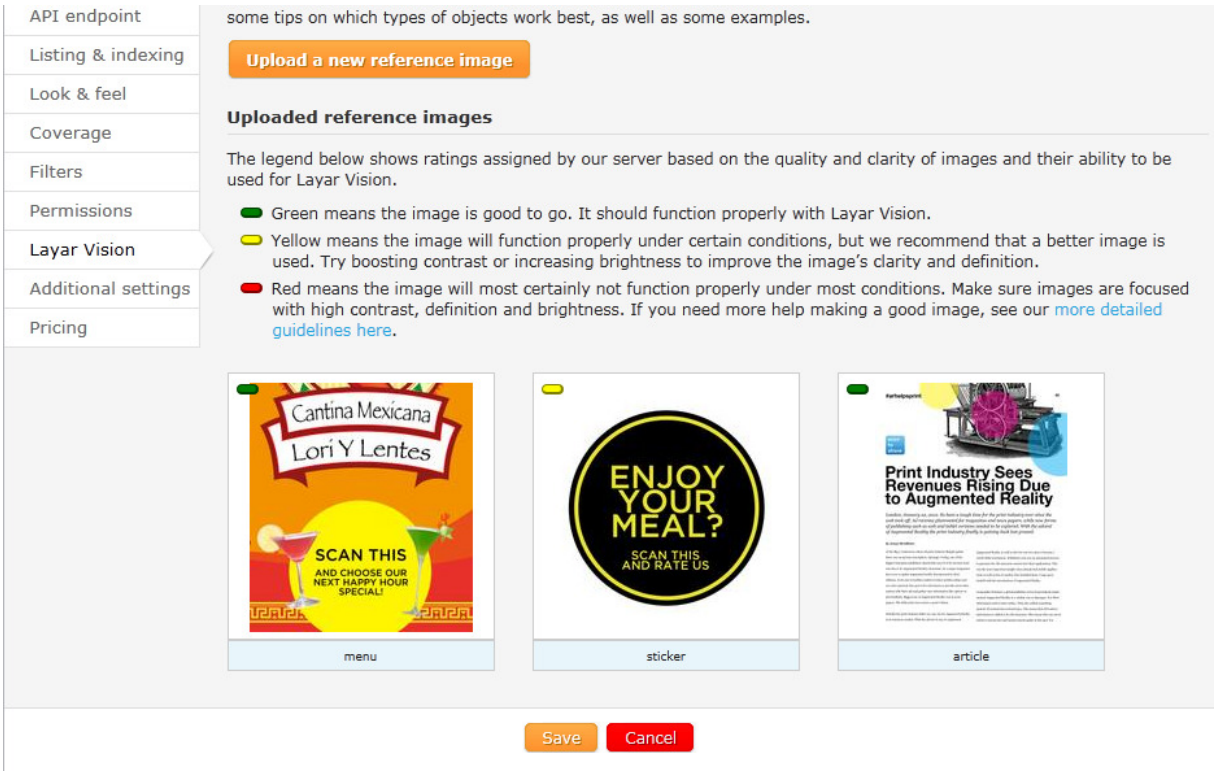
Step 3 – Enabling Layar Vision

The last remaining task to get the new Layar Vision technology to actually work with your layer is uploading the reference images (can be found under "Reference Images" directory within the Archive folder).

A reference image is the image that the Layer Vision technology will try to spot when running your layer. Once you upload the reference image the server will create a unique fingerprint that allows the Layar app to actually "see" the image in its surroundings.

To upload the reference images make sure you're editing your layer in the Publishing Environment and click on the "Layar Vision" tab. Now, upload the reference images that are in the archive attached to this document and make sure to use the same Image Keys as in the screenshot below.

Once the three images are uploaded and done processing, click "save".



Congratulations! You are now ready to test your layer.

Opening your Layer

Now you should be able to find your layer in the Layer app.

Make sure to login to Layar using your developer account and look for your handywork under Layers→Test→<Your Layer's name>

If you want, you can print out the images in the directory "Images for print" and use them to test you layer. There's no need to print them in full color as grey scale will work just as well.
More environment friendly is to just view the images on your computer display but due to the reflection of the display the results might be a bit shaky.

What makes it tick

Below you can find the snippets of the PHP script's output together with explanation.

As mentioned before, the getPOIs response the PHP code outputs is formatted as JSON.

The minimal response, with only the required fields and without any hotspots is quite simple:

```
{
   "layer": "cantine_docs", // The name of your layer
   "hotspots": [], // an empty hotspots array. Not required.
   "errorCode": 0, // No error happened on the PHP's side, so no error code
   "errorString": "ok" // No error, so all is "ok"
}
```

Obviously, this response doesn't really do much but it also won't produce any errors so to make things interesting, hotspots (Points Of Interest) need to be added.

Scan to share

Let's first look at the "Scan to Share" functionality. When you hover over a print of the newspaper article included in the archive's "Images for print" directory a simple but that reads "Scan to share" shows up. When clicking that button, the layer sharing screen is shown.

This is all triggered by the following hotspot

```
{
        "id": "1_1", // This is a unique ID within this layer.
        "showBiwOnClick": false, // Don't show the BIW when clicking the POI.
        "showSmallBiw": false, // Don't show the small BIW (when spotting a POI).
        // Next is the text object in which you can specify the contents of the BIW.
        // Since the BIW is hidden, only the required field "title" is present.
        "text": {
                "title": "Bogus title"
        },
// The achor object tells the client whether we're dealing with a POI that has a specific geolocation or a reference image.
// In this case we're obviously dealing with a reference image.
        "anchor": {
                "referenceImage": "article" // The "article" value refers to the image we uploaded in the Layar Publishing Environment before.
        },
// The object object tell Layar what type of object should be displayed for this POI. 3D objects can be used but in this case we simply use a 2D png image.
        "object": {
                "contentType": "image/png",
                "url": "http://layar.vendite.nl/cantine/assets/POI_sts.png",
                "size": 0.04 // The size of the augmentation in meters. This size is calculated relatively to the size of the reference image, which size is defined in the Layar Publishing Environment.
        },

// The transform object allows you to configure the displayed object's behavior.
// Once can reposition the object relatively to the reference image, scale it, rotate it or make it always face the user.
        "transform": {
                "rotate": {
                        "rel": false, // Always have the POI facing the user? No.
                        "axis": {
                                "x": 1,
                                "y": 1,
                                "z": 1
                        }
                },
```

```
                "scale": 1 // Telling the client to not do any scaling. To leave it at 100% which happens
to be 0.04 meter (defined in object.size)
        },
//The actions array defines the actions that are possible for this POI. Normally the available actions for
a POI would be triggered by the user by clicking the desired action in the BIW.
// However, we disabled the BIW as you have seen a couple of lines earlier, so there's no way for the user
to trigger that action.
// When you only define 1 action and you hide the BIW the action triggers automatically wheb clicking the
POI.
// This action's uri uses the "layarshare" protocol, which triggers the sharing dialog in Layar.
        "actions": [
                {
                        "uri":
"layarshare://cantine/?description=Print%20Industry%20Sees%20Revenues%20Rising%20Due%20to%20Au
gmented%20Reality&title=%23arhelpsprint&type=message&link=http://www.layar.com/browser/layar-
vision/",
                        "label": "Share!", //The label is mandatory but won't be shown since the BIW is
not being shown
                        "contentType": "application/vnd.layar.internal"
                }
        ]
}
```

Used API features in this demo:

- Actions
- Sharing API

Rate us:

At first sight you might expect that the Cantine layer, which uses only 3 reference images, only has 3 hotspots, but when you load the layer you may have noticed Layar actually finds 12 results.

That is because every item on the screen that has a different and/or independent behavior needs to be specified separately.

A good example of that fact is the "Rate us" sticker that shows a thumbs up and a thumbs down, augmented at the sides of the sticker. Both thumbs are separate POIs which have their own behaviours when clicked up.

Have a look at the following 2 hotspots objects:

```
{
    "id": "3_1",
    "text": {
        "title": "Bogus title"
    },
    "showBiwOnClick": false,
    "showSmallBiw": false,
    "anchor": {
        "referenceImage": "sticker"
    },
    "transform": {
        "translate": {
                "x": -0.080,
                "y": 0,
                "z": 0
        },
        "scale": 0.5
    },
    "actions": [{
            "uri": "layar://cantine/?action=refresh&rate=good",
```

```
            "label": "Share!"
        }
    ],
    "object": {
        "url": "http:\/\/layar.vendite.nl\/cantine\/assets\/thumb_up.png",
        "contentType": "image/png",
        "size": 0.16
    }
},
{
    "id": "3_2",
    "text": {
        "title": "Bogus title"
    },
    "showBiwOnClick": false,
    "showSmallBiw": false,
    "anchor": {
        "referenceImage": "sticker"
    },
    "transform": {
        "translate": {
                "x": 0.080,
                "y": 0,
                "z": 0
        },
        "scale": 0.5
    },
    "actions": [{
            "uri": "layar://cantine/?action=refresh&rate=bad",
            "label": "Share!"
        }
    ],
    "object": {
        "url": "http:\/\/layar.vendite.nl\/cantine\/assets\/thumb_down.png",
        "contentType": "image/png",
        "size": 0.16
    }
}
```

You may have noticed that both objects are practically equal. The only differences are that they are:

- Positioned differently
- Contain different URL's for their actions

Visually this could have been solved by using only 1 POI with a bigger, transparent png image containing both thumbs, but from a functional point of view this would not allow us to vote either positive or negative.

This example also demonstrates how you can link more than one POI to 1 reference image.

Margaritas

The margaritas require a bit more POIs.

There's one for the caption at the top. We can choose to add a POI with the text in a transparent image or 3D object. Obviously using an image is the simplest approach.

Each Margarita glass has it's own POI too. For the glasses 3D objects were used which were made with Blender and then converted into Layers own l3d format using the "Layar3DModelConverter".

At the foot of every glass there's another POI containing the name of the margarita.

So, in total 9 POIs were used to make up the "scene".
All of the POIs use the same reference image but are positioned differently relative to the reference image.

Because of the size of the entire hotspot objects only the first 3 are pasted below.

The textual header (id: 2_0), the first glass (id: 2_1) and it's name at the foot of the glass (id: 2_1-strawberry).


```
{
    "id": "2_0",
    "text": {
       "title": "Bogus title"
    },
    "showBiwOnClick": false,
    "showSmallBiw": false,
    "anchor": {
       "referenceImage": "menu"
    },
    "transform": {
       "rotate": {
           "rel": false,
           "axis": {
              "x": 1,
              "y": 0,
              "z": 0
           },
           "angle": 45
       },
       "translate": {
           "x": -0.01,
           "y": 0.035,
           "z": 0.045
       },
       "scale": 0.016
    },
    "object": {
       "url": "http://layar.vendite.nl/cantine_docs/assets/margaritas/choosefav.png",
       "contentType": "image/png",
       "size": 1
    }
},
{
    "id": "2_1",
    "text": {
       "title": "Bogus title"
    },
    "showBiwOnClick": false,
    "showSmallBiw": false,
    "anchor": {
```

```
        "referenceImage": "menu"
    },
    "transform": {
        "rotate": {
            "rel": false,
            "axis": {
                "x": 0,
                "y": 0,
                "z": 1
            },
            "angle": 45
        },
        "translate": {
            "x": -0.04,
            "y": 0.01,
            "z": 0
        },
        "scale": 0.012
    },
    "actions": [
        {
            "uri":
"http://layar.vendite.nl/cantine_docs/webcontent/index.php?margarita=strawberry",
            "label": "Share!",
            "contentType": "text/html",
            "activityType": 1,
            "showActivity": true
        }
    ],
    "object": {
        "url": "http://layar.vendite.nl/cantine_docs/assets/margaritas/drink_red_60.l3d",
        "size": 4,
        "contentType": "model/vnd.layar.l3d"
    }
},
{
    "id": "2_1-strawberry",
    "text": {
        "title": "Bogus title"
    },
    "showBiwOnClick": false,
    "showSmallBiw": false,
    "anchor": {
        "referenceImage": "menu"
    },
    "transform": {
        "rotate": {
            "rel": false,
            "axis": {
                "x": 1,
                "y": 1,
                "z": 1
            },
            "angle": 0
        },
        "translate": {
            "x": -0.04,
            "y": 0.005,
            "z": 0
        },
        "scale": 0.5
    },
    "actions": [
        {
            "uri":
"http://layar.vendite.nl/cantine_docs/webcontent/index.php?margarita=strawberry",
            "label": "Share!",
```

```
            "contentType": "text/html",
            "activityType": 1,
            "showActivity": true
        }
    ],
    "object": {
        "url": "http://layar.vendite.nl/cantine_docs/assets/margaritas/strawberry.png",
        "contentType": "image/png",
        "size": 0.07
    }
}, …
```

Next steps

The next step is up to you. You could start your own layer now or you could extend this layer a bit further.

Just for fun, try to make the margaritas rotate in one direction while their labels rotate in the opposite direction. (Hint: animation.onCreate)

And if that's not challenging enough: Try animating the thumbs so they float out of the screen once a thumb has been clicked. (Hint: animation.onDelete)

More information about animation API can be found here.

Layar offers developers the building blocks to do practically anything from building strategy games where you play real people to adding the AR experience to printed media.

Enjoy!