# **Dynamic Tile Content**

DEVU project Name - TEST\_KOVAION

- TEST\_APP\_PKG\_KOV

## **Introduction**

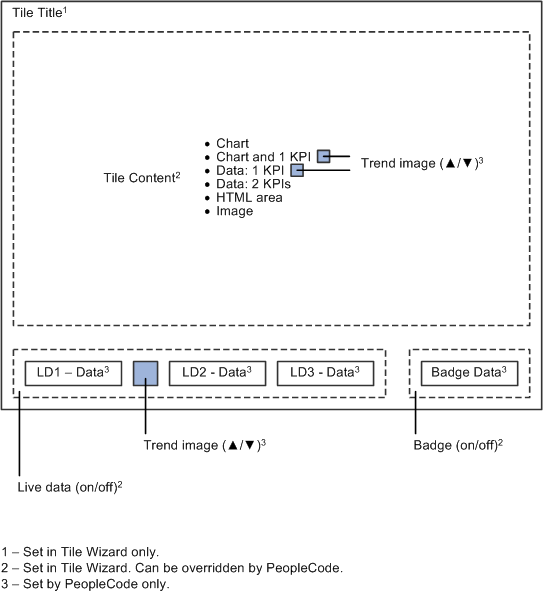
We can show dynamic data on tile by specifying an application class as the data type in Tile Wizard. In addition to specifying the application class ID, we also specify these options in Tile Wizard:

**Tile content type**

**Live data (on/off)**

**Badge data (on/off)**

These options determine what dynamic content and data will appear in the tile. Our implementation of the application class in PeopleCode will determine the specifics of the dynamic content and data. However, our PeopleCode implementation can also override the options chosen in Tile Wizard.



1. **Tile Title -** The tile’s title appears in the title bar at the top of the tile. The title is defined at Step 1 in Tile Wizard, and cannot be overridden in PeopleCode.
2. **Tile Content -**The tile content consumes the majority of the face of the tile, beneath the title bar and above the live data and badge areas on the tile. While the tile content type is specified at Step 2 in Tile Wizard, it can be overridden in PeopleCode. The specific tile data for any tile content type must be defined in PeopleCode.
3. **Live Data -** The live data region appears at the bottom of the tile. Whether live data is displayed on a tile is specified at Step 2 in Tile Wizard; this setting can be overridden in PeopleCode.

Live data consists of four elements, each of which is optional:

**Live data value 1 + Trend image + Live data value 2 + Live data value 3.**

## **Creating Application Package for the Dynamic Tile**

To implement an application class for tile content, do the following in your application class definition:

1. Import the base class (also referred to as the superclass):

import PTGP\_APPCLASS\_TILE:Tiles:Tile;

Notes: Make sure the App Package contains a Sub Package. Otherwise, we might run into errors.

1. In the class declaration, do the following:
2. Indicate that your class (the subclass) extends the base class.

class TileD1 extends PTGP\_APPCLASS\_TILE:Tiles:Tile

1. Declare the constructor method for your class.

method TileD1();

1. Declare the required getTileLiveData method.

method getTileLiveData();

1. (Optional) Declare any additional private methods and properties required by your implementation.
2. In the constructor method for your class, instantiate an object of the superclass:

method TileD1

%Super = create PTGP\_APPCLASS\_TILE:Tiles:Tile();

end-method;

1. In your implementation of the required getTileLiveData method, do the following:
2. (Optional) Override the content type by invoking one of the optional SetTileContentAs\* methods.

Type of tile content (\*):

- Chart - Chart and 1KPI (key performance indicator)

- 1 KPI - 2 KPIs

- HTML - Image

1. (Optional) Override whether live data is displayed on the tile by invoking the setTileLiveData method or setting the hasLiveDataDescr property.
2. (Optional) Override whether badge data is displayed on the tile by invoking the setTileHasCount method or setting the hasLiveDataCount property.
3. For the tile content type selected, invoke any require methods and set all required properties for that content type.
4. If live data is enabled for the tile, set all required properties for live data.
5. If badge data is enabled for the tile, set all required properties for badge data.
6. (Optional) Implement error handling by setting the hasContent property.
7. Create a tile definition in Tile Wizard. At Step 2, specify your custom application class.

## **Defining Application Class and Tile Content**

### **Defining the Application Class when the Tile content is Image**

Invoke the SetTileContentAsImage method within your implementation of the getTileLiveData method only if you wish to dynamically override the tile content type to display an image at runtime. When the tile content is specified to be an image, either setTileImageRef method or the ImageReferenceField property is required.

When the user want to use image as tile content, the following methods and property can be used to define the Tile Content.

#### **Methods**

* + **SetTileContentAsImage()-** invoked in getTileLiveData() to dynamically override the tile content type at runtime.
  + **setTileImageRef**(image\_name)

This method is used to specify the image to be displayed when the tile content is hosen as image.it is equalivalent to specifying the value for property ‘ImageReferenceField’. Either of the two can be used when the tile content is specified as image.

%This.setTileImageRef(IMG\_NAME);

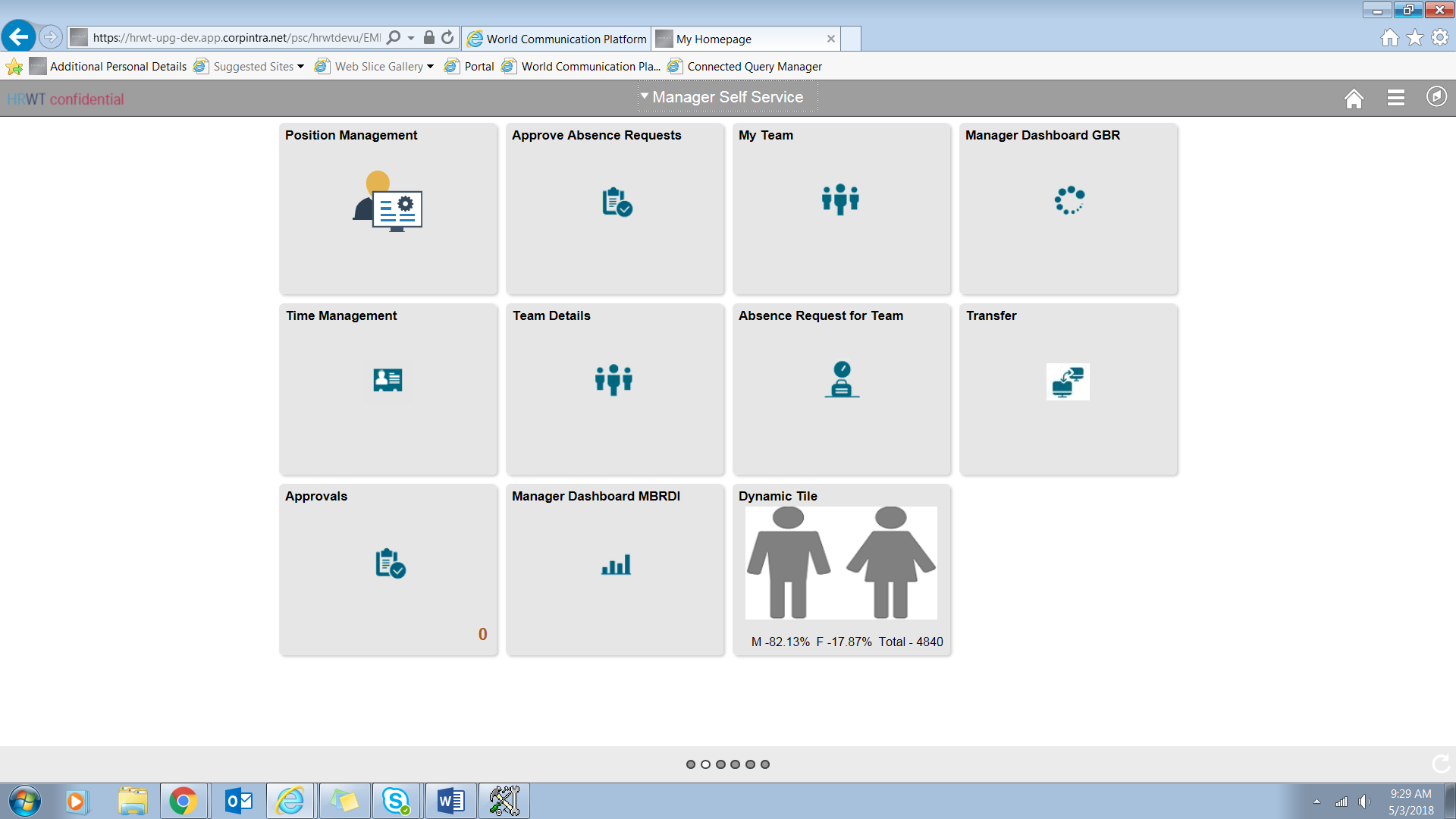
#### **Properties**

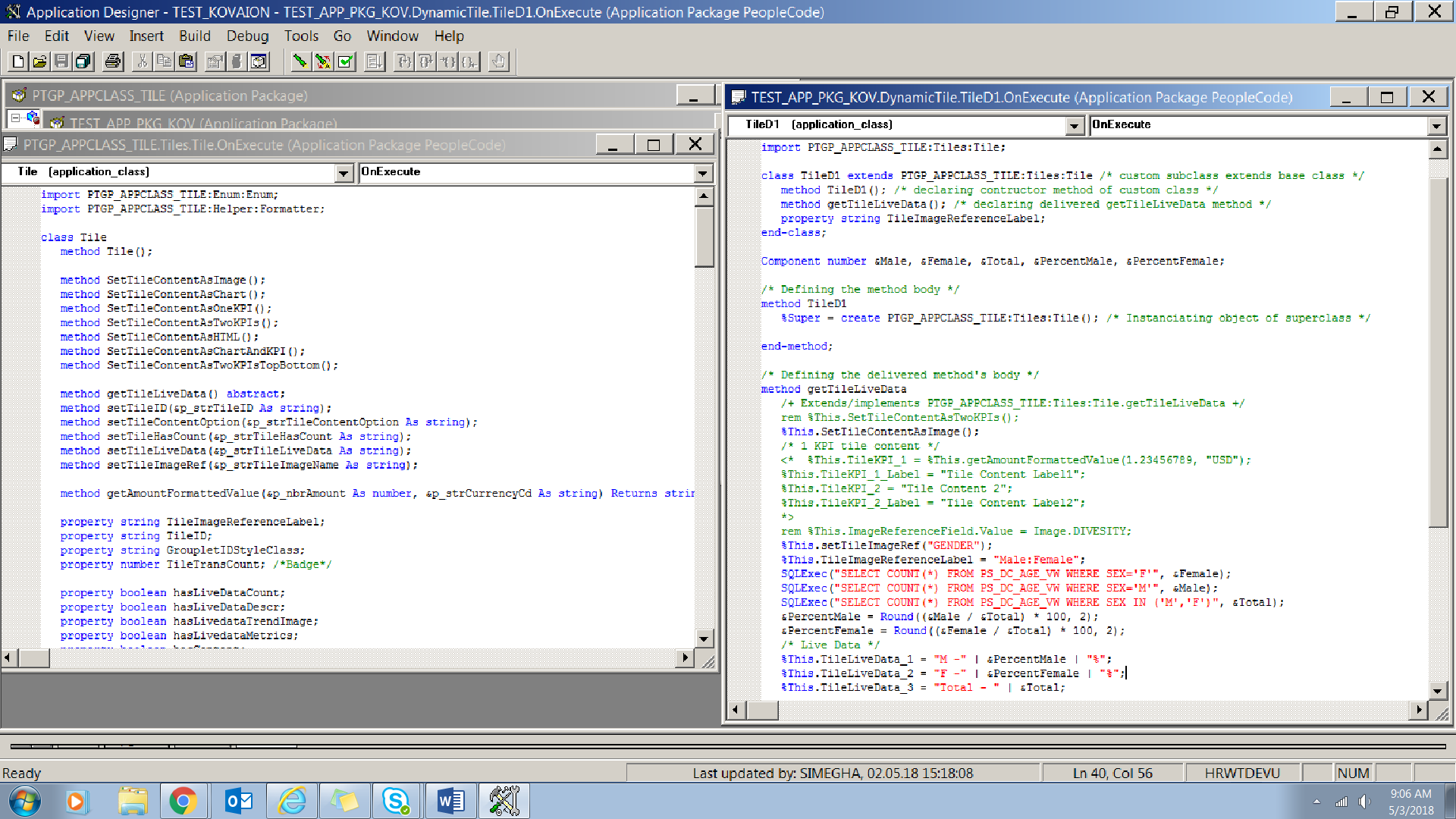
* + **ImageReferenceField –** It is used to specify the image field property when the tile content is specified as image. As it is a Field object , we can give value property as follows:

%This.ImageReferenceField.Value=Image.IMG\_ID;

* + **TileImageReferenceLabel –** This property is used to set tooltip (Hovertext) for the image set as tile content.It is visible by the user when the image is hover over.

%This.TileImageReferenceLabel(“This is a image hover text”);





## **Setting Live Data**

Live Data is displayed at the left bottom of the tile. 3 Live data can be displayed on the dynamic tile. Live data can be enabled from Step 2 of Tile Wizard or we can also override it through PeopleCode.

Live data consists of four elements, each of which is optional:

**Live data value 1 + Trend image + Live data value 2 + Live data value 3**

### **Method and Properties used to define Live Data**

**hasLiveDataDescr** – This property is used to enable the Live Data area. It returns a Boolean value. It is equivalent to setTileLiveData method.

%This.hasLiveDataDescr= “True”;

**setTileLiveData**

%This.setTileLiveData(“Y”); /\* Enables the Live Data Area \*/

## **Setting Badge Data**

It is the data which appear at the botton right of the tile , right of Live Data.It is a simple interger, typically a count of item.

First we need to enable the badge area (whether the user want the badge area to be displayed).This can be done using delivered method as well as property.

#### **Method**

* + **setTileHasCount**(badge\_display – Y/N)

Y – Badge area is enabled.

N – Badge area is disabled.

We invoke this method in ‘getTileLiveData’ to override whether the badge area should enabled or not. We can enable it using hasLiveDataCount property also. After the badge area is enabled , the bdage data needs to be set through TileTransCount Property.

%This.setTileHasCount("Y"); /\* Enables the badge area. \*/

%This.TileTransCount = MY\_REC.COUNT;

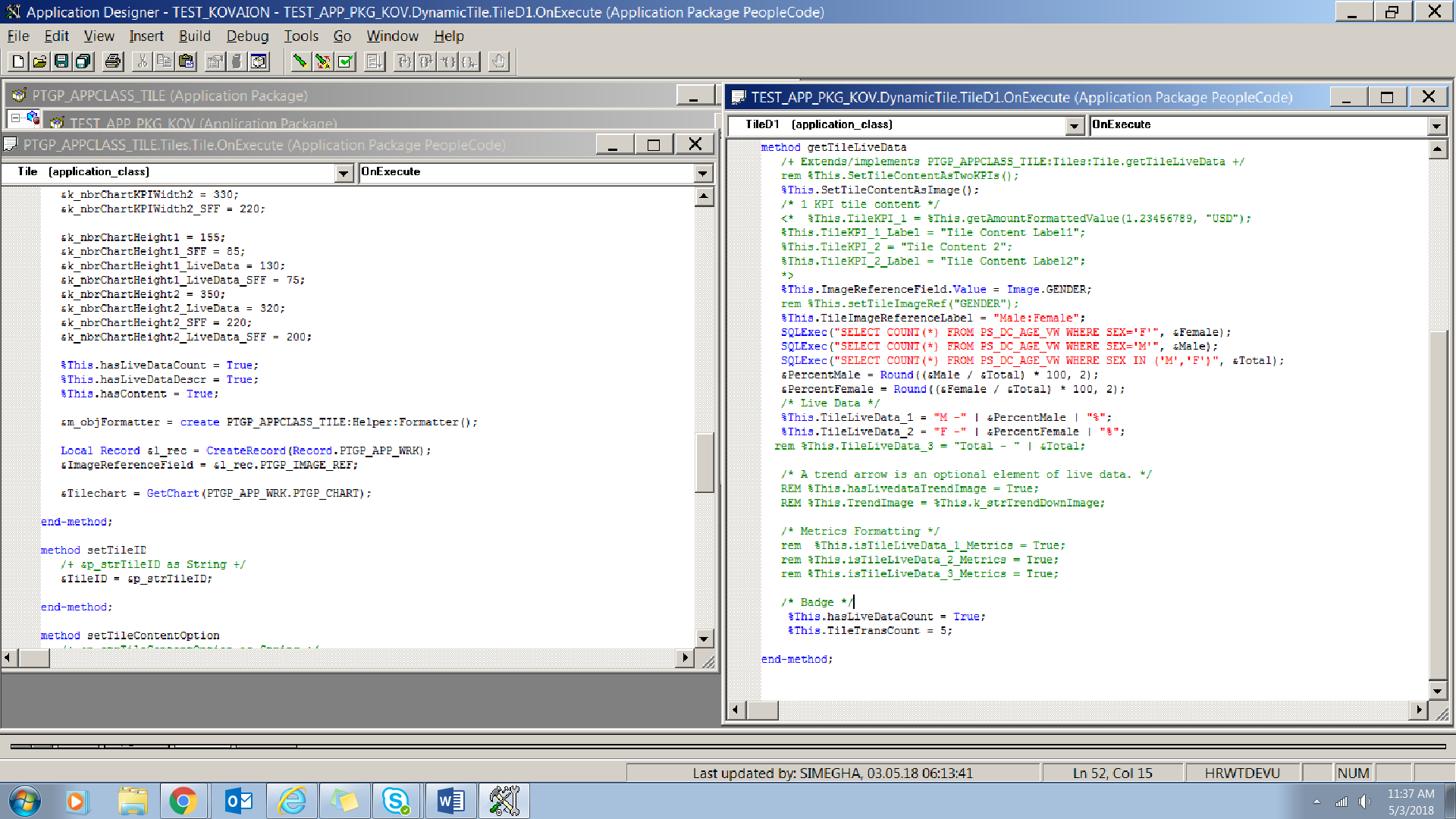
#### **Properties**

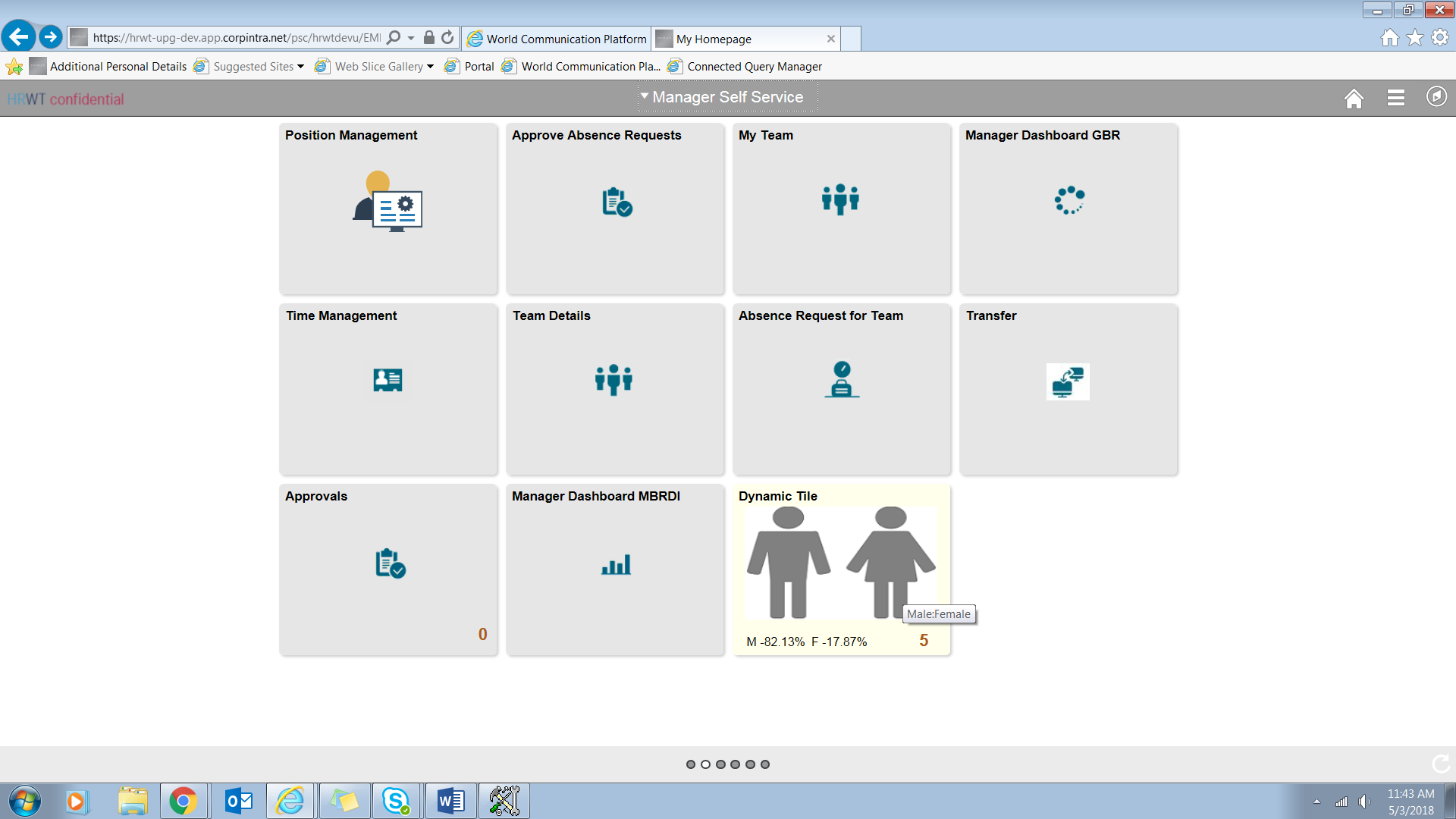
* + **hasLiveDataCount -** Use the hasLiveDataCount property to set or return a Boolean value indicating whether the badge area is enabled.

%This.hasLiveDataCount = True; /\* Enables the badge area. \*/

%This.TileTransCount = MY\_REC.COUNT;

* + **TileTransCount –** this property is used to set the badge data when the badge area is enabled.

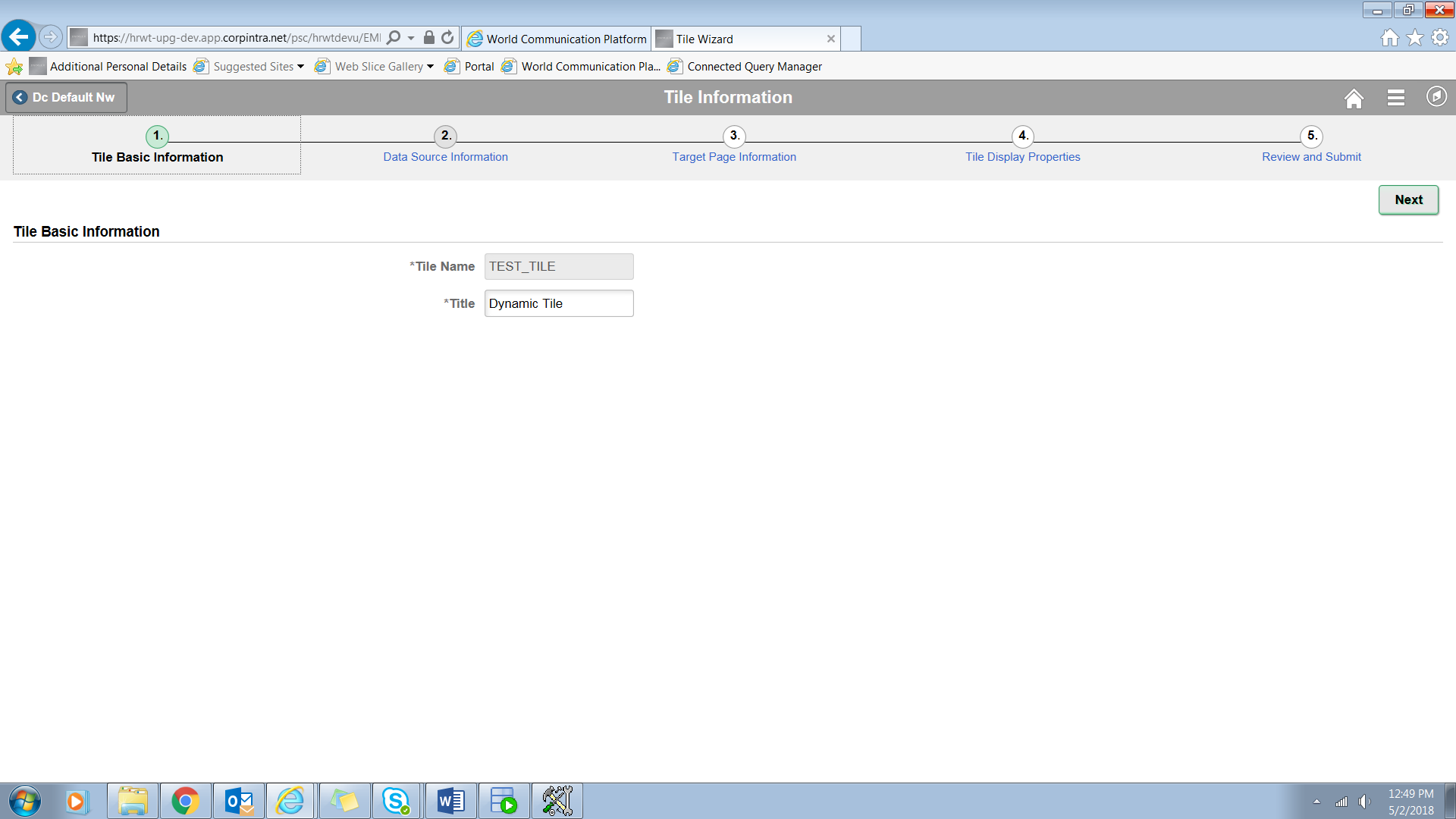




## **Creating a tile definition in Tile Wizard**

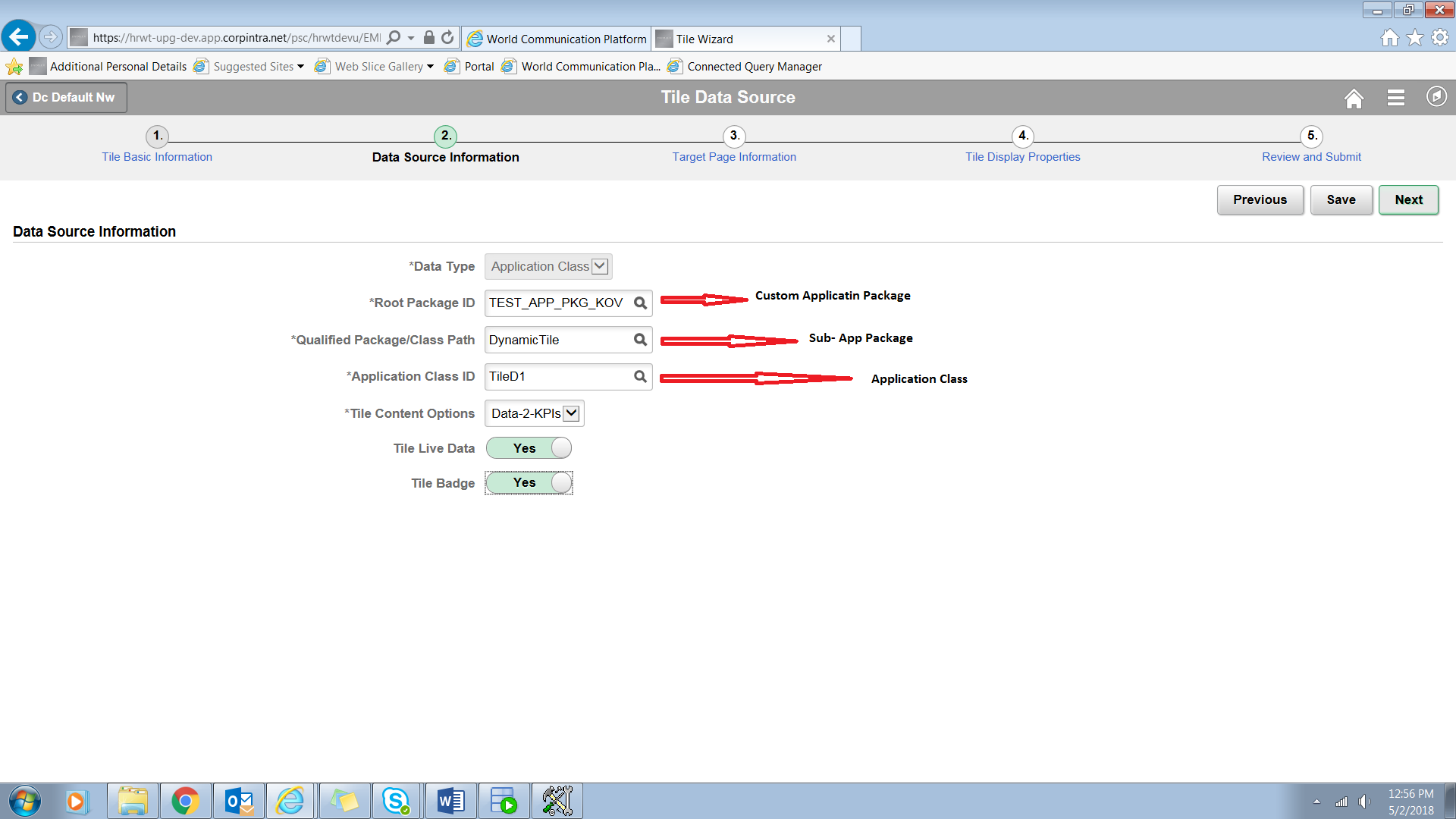
**Navigation : Navigator - > PeopleTools - > Portal - > Tile Wizard**

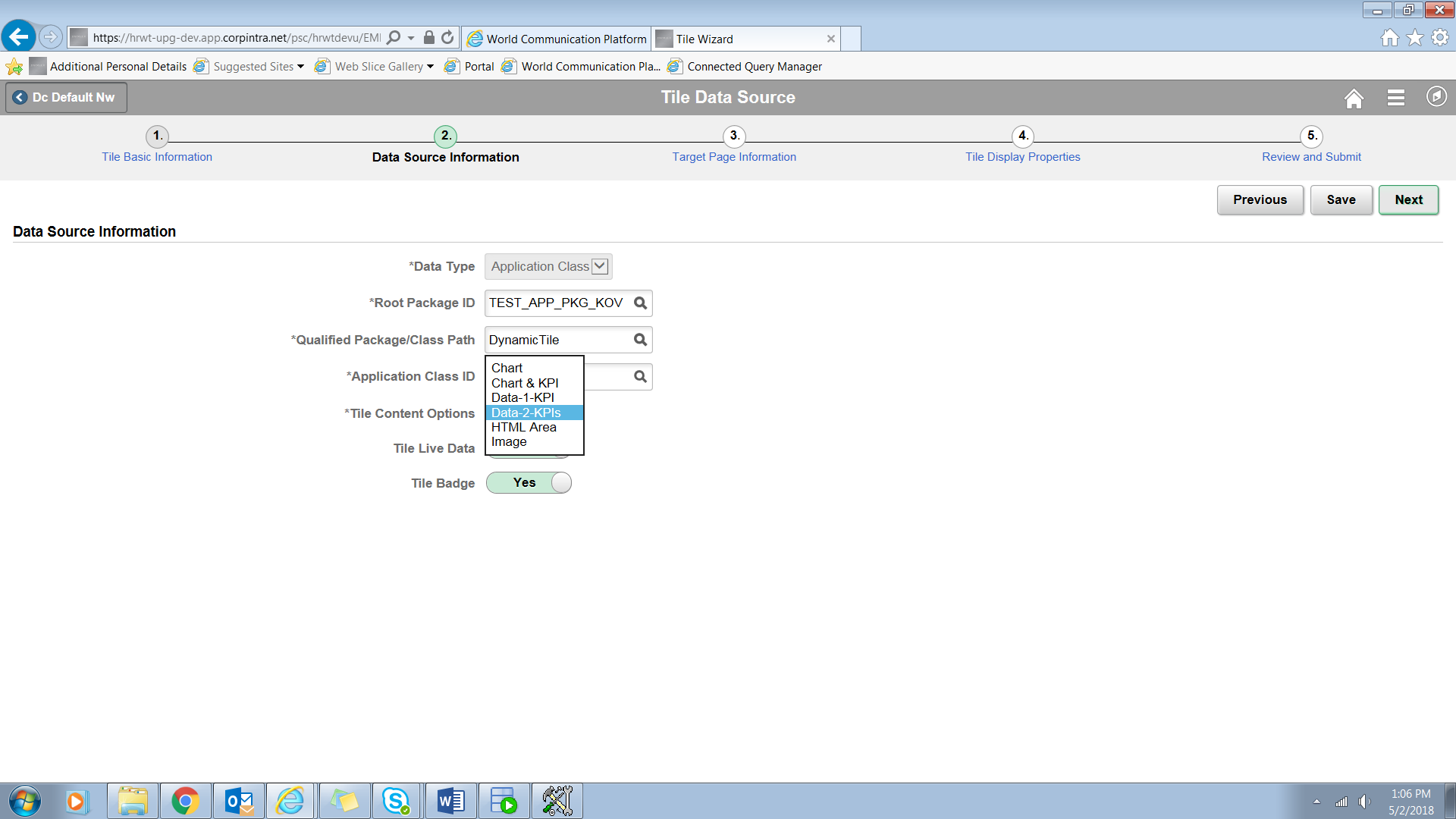
**Step 1:** Give the tile Name and its Title and click on Next.



**Step 2:** Specify the Data Source for the Tile Wizard .Select the Application Package , Sub-package and Class in this step. We also specify the tile content (chart , chart&KPI, 1KPI etc.) along with whether the tile has Live Data and Badge or not.

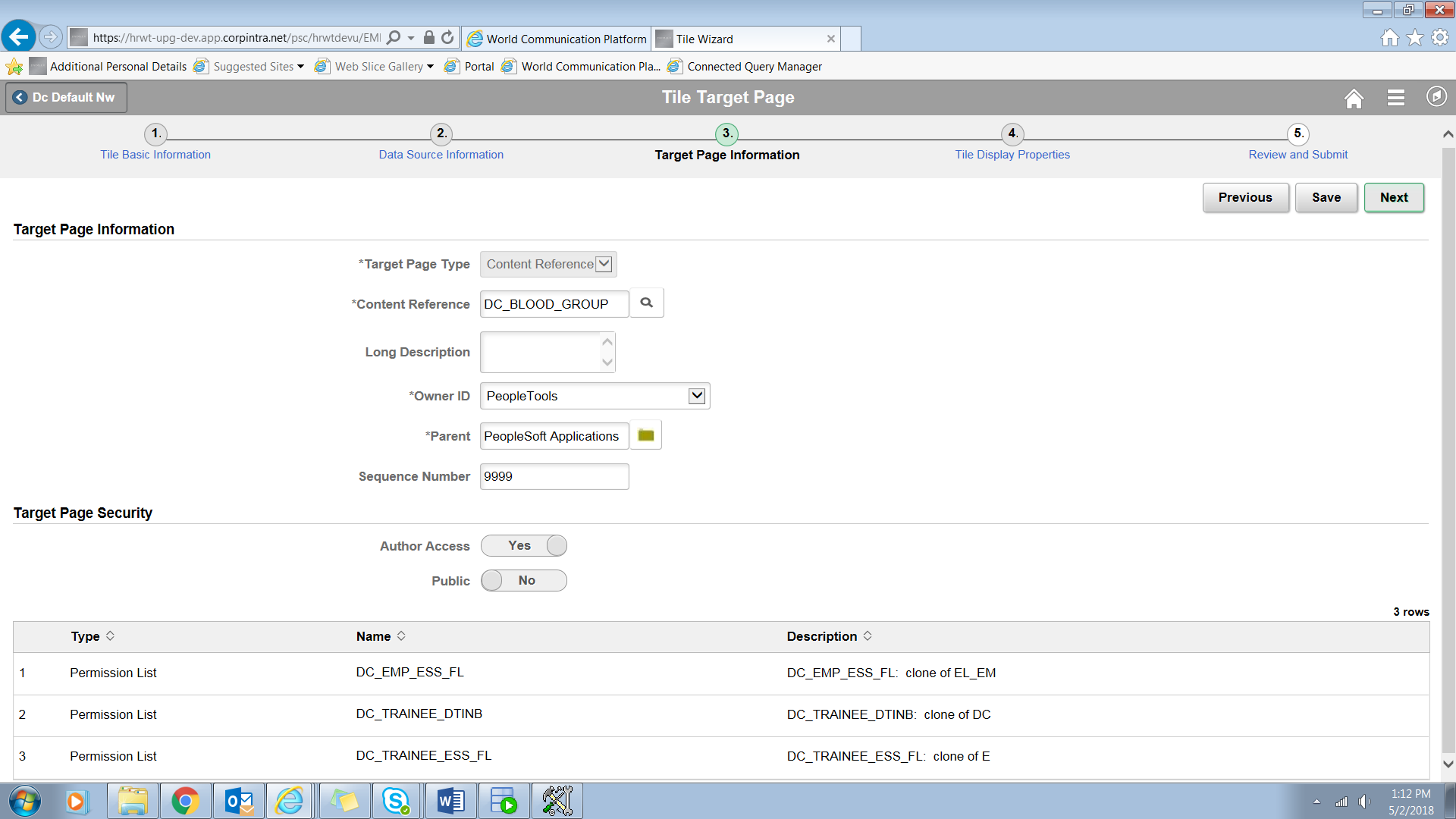
1. **Root Package ID** - Select a custom application package that includes the class that implements the PTGP\_APPCLASS\_TILE:Tiles:Tile base class and one of its required SetTileContentAs methods.
2. **Qualified Package/Class Path -** Select the package name or package path to the implemented class.
3. **Application Class ID** - Select the class that implements the required getTileLiveData method.





**Step 3:** Specify the Content Reference which you want see when the tile is tapped. Choose it through the prompt given beside the field. Give the Owner Id and Parent folder being mandatory.

You can provide security to the tile of who can access and who can not by selecting the required Permission & Role for the tile.



**Step 4:** Specify the width and height of the tile which could be (1:1, 1:2, 2:1, 2:2). An image for it which could be overided through PeopleCode.

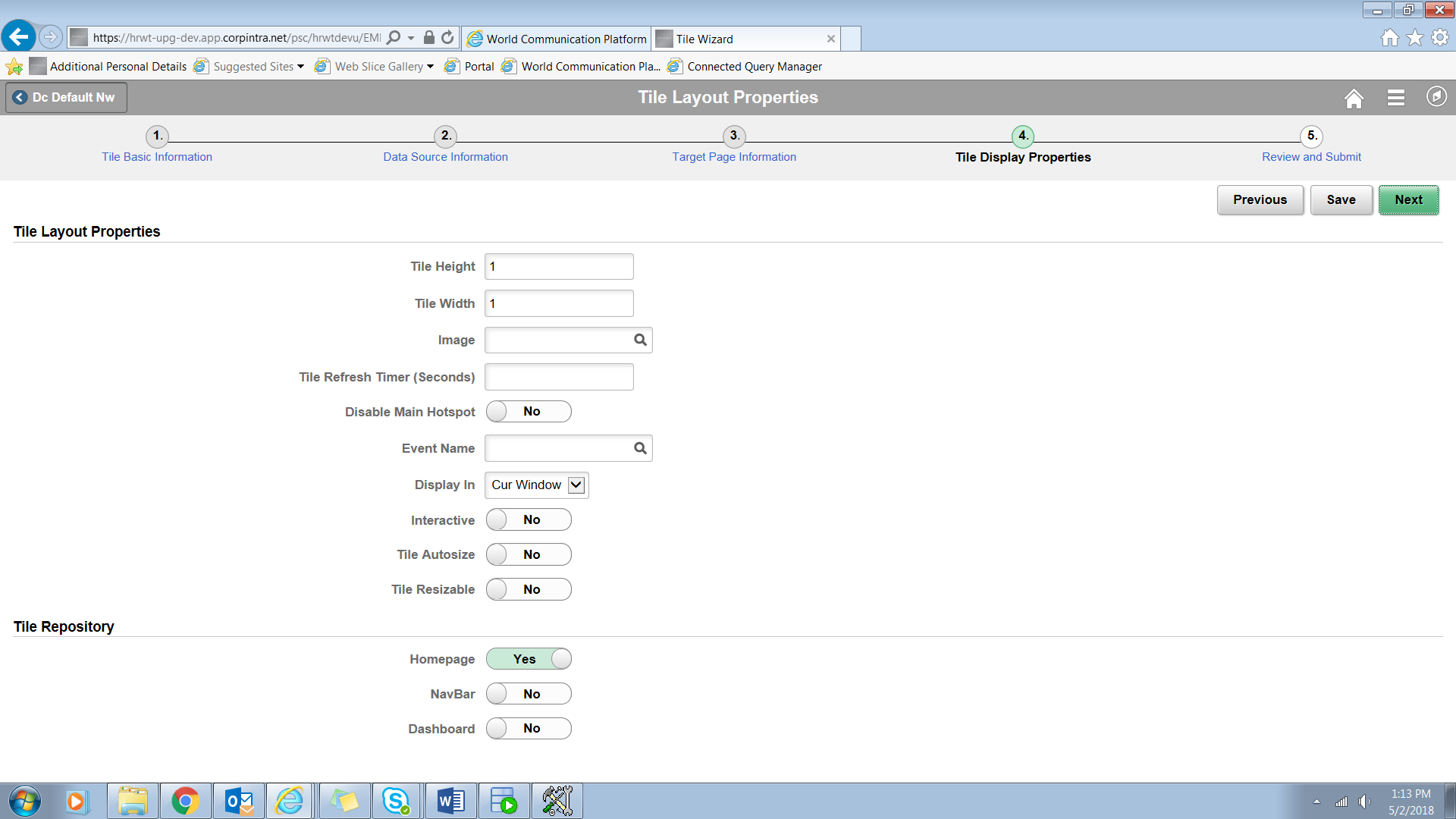
1. **Image -** Select an image (in SVG format only) from the database to display a custom static image as the fluid content. Otherwise the default image is displayed.
2. **Tile Refresh Timer (Seconds) -** Enter the time in seconds to set an automatic refresh period for dynamic content on a tile. When the timer limit has been reached, the system re-draws the tile so that it displays the current data, such as in the case with chart.

**Note:** If any Event Name is selected this field is disabled. Also the system enforces a 10 second minimum limit. Any value entered less than 10 seconds is ignored and is treated as 10 seconds.

The default value of 0 disables any automatic refresh.

1. **Disable Main Hotspot -** Select this option to ‘Yes’ to disable displaying the target content defined for the tile when the tile is tapped. However, any links displayed dynamically (dynamic data /content) within the tile are not disabled.
2. **Display In –** Control how the content appears once a user taps n the tile.
3. Cur Window – target content opens up in current window.
4. Modal – target content opens up in model window. We get extra Modal Parameter to specigy on select this option.
5. NavBar – target content appears in NavBar only when the tile is added in NavBar else it will open in the current window.
6. New Window – target content opens up in a new window.
7. **Interactive -** Select this option to display the fluid content as interactive, which allows users to enter data or click buttons within the fluid content.

It is disabled for Pivot grid tiles and OBIEEE tiles.



**Step 5:** Review all the settings given for the tile in the last step and Publish the Tile Wizard.

