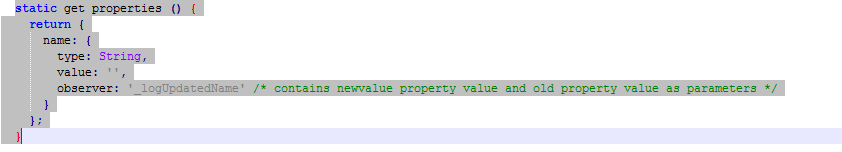
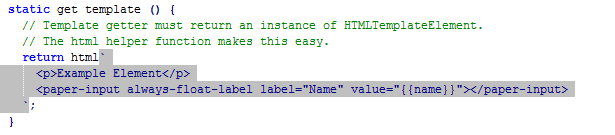
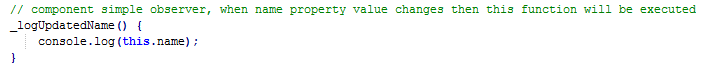
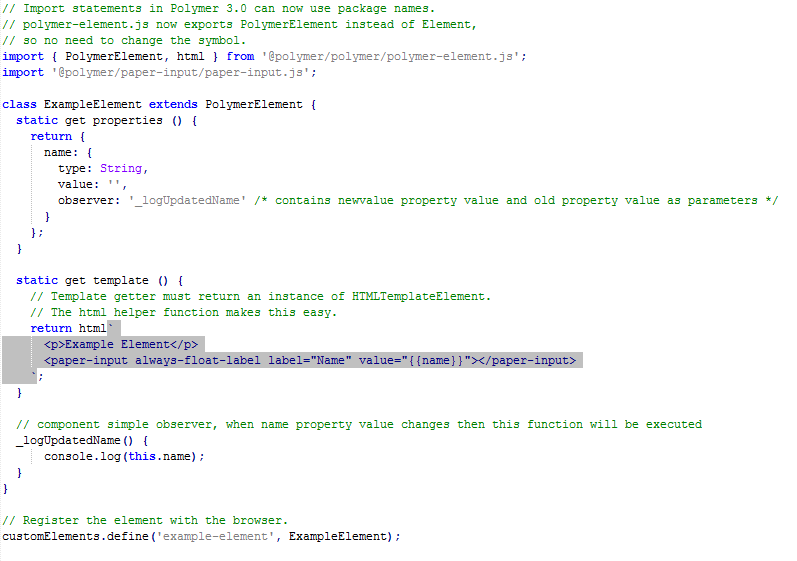
# Polymer

## Concepts:

* [Component creation](#_Component_creation,_Creating)
* [Creating properties](#_Component_creation,_Creating)
* [Property Observers](#_Component_creation,_Creating)
* [Routing](#_Routing:)
  + [iron-selector](#_iron-selector:)
  + [iron-pages](#_iron-pages:)
  + [\_routePageChanged observer (app-route observer)](#___routePageChanged_observer)
  + [\_pageChanged observer (component observer)](#__pageChanged_observer_(component)
* [Navigation from components](#_Navigation_from_components:)
* [Component communication](#_Component_communication:)
  + [Parent to child communication](#_Parent_to_Child:)
  + [Child to parent communication](#_Child_to_Parent:)
  + [Event triggering from child component to parent component](#_Event_triggering_from)
* [If condition using dom-if](#_If_condition_using)
* [Loop array items using dom-repeat](#_Loop_array_items)
* [Lifecycle callback’s](#_Lifecycle_callback’s:)
* [Global variables](#_Global_variables:)
* [Loading External css globally](#_Loading_External_css)

## Component creation, Creating properties, Property observers:

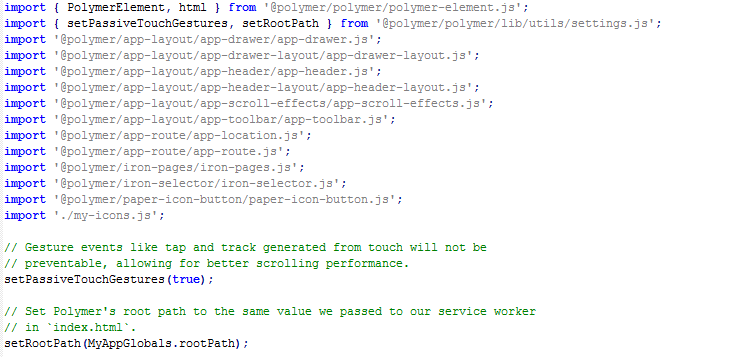
* Go to src/ folder
* Create new js file. Example:
  + example-element.js
* **Create component properties at properties function:**
* 
* **Create html tags inside template:**
* 
* **Component property simple observer:**
* 
* **Total example component looks like this with html, properties, simple observer, import statements:**



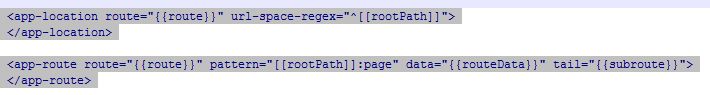
## Routing:

As example of myapp.js routing

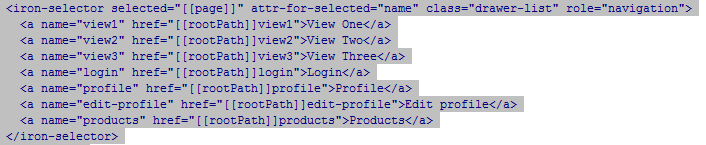
### Import related js files in my-app.js and set RoutePath:



### <app-route> and <app-location> elements:



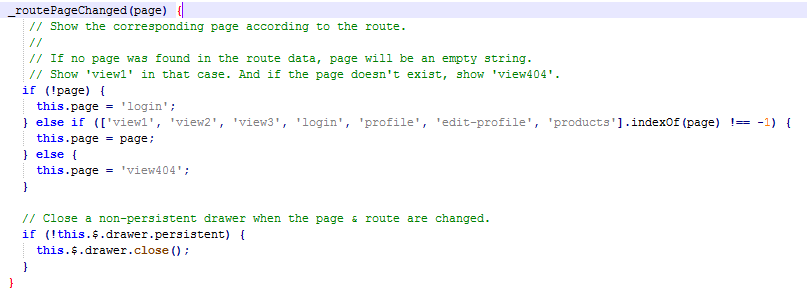
### iron-selector:



### iron-pages:



### \_ routePageChanged observer (app-route observer):

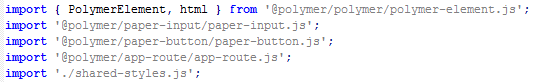


### \_pageChanged observer (component property observer):



## Navigation from components:

### Import app-route component:



### Set app-location:



### Set route-path:



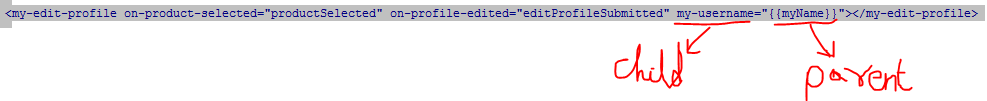
## Component communication:

Example of components communication with **myUsername** property.

### Parent to Child communication:

Inside parent component, we are setting **my-username = {{mName}}** at child component element.

Whatever we set data to **myName** property will be passed to **myUsername** property inside my-edit-profile component.

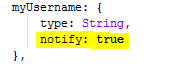




### Child to Parent communication:

Inside child component property, use **notify: true** inside property to set updated data to parent component

Whatever we change data to **myUsername** property at child component will be updated at **myName** property inside parent component. (Two way binding)

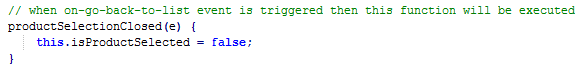




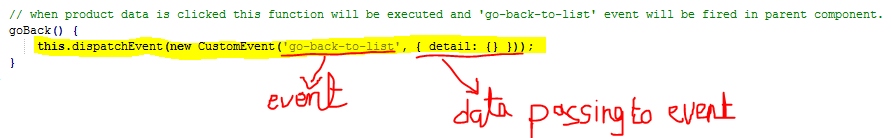
### Event triggering from child component to parent component:

Inside parent component:





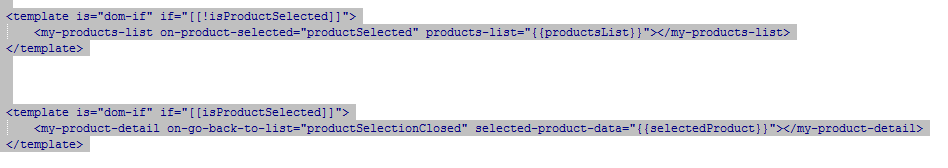
From child component, trigger event to parent component:



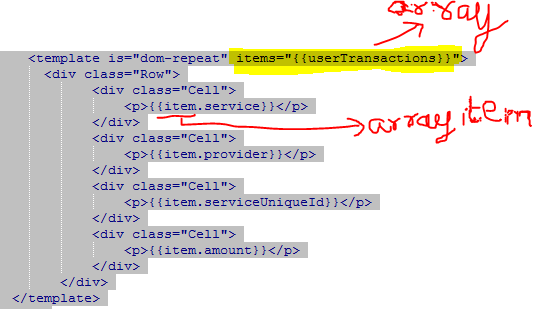
## If condition using dom-if:

### Import dom-if js file and set if condition using template tag:





## Loop array items using dom-repeat:



## Lifecycle callback’s:

### Ready Callback:

ready(){

super.ready();

}

### Constructor:

constructor() {

super();

}

## Global variables:

Don’t know how to set global variables.

## Loading External css globally:

Don’t know how to set external css file.