

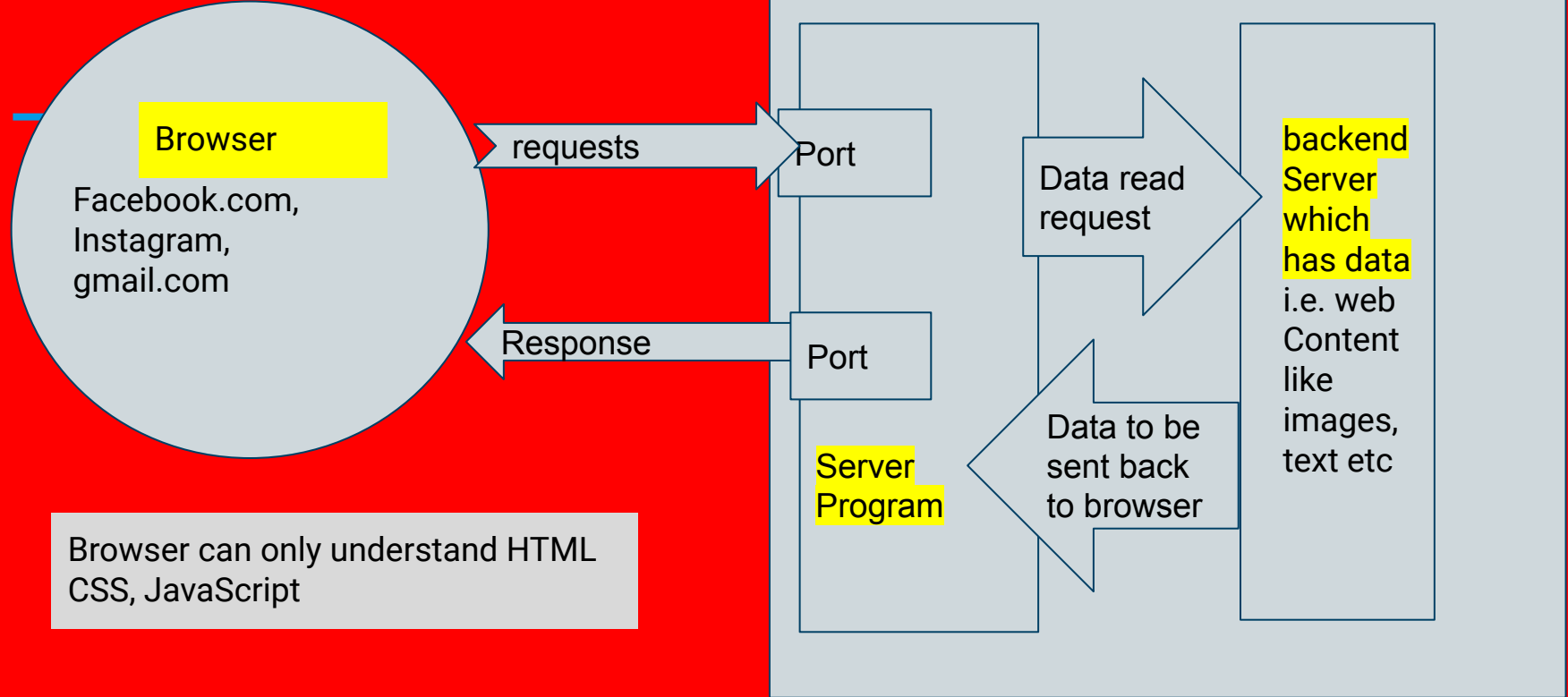
SAPUI5

Satya Kondiparthi

User Interface(UI) Programming Toolkit from SAP, to build SAP business web applications. FIORI Applications are also built on SAPUI5.

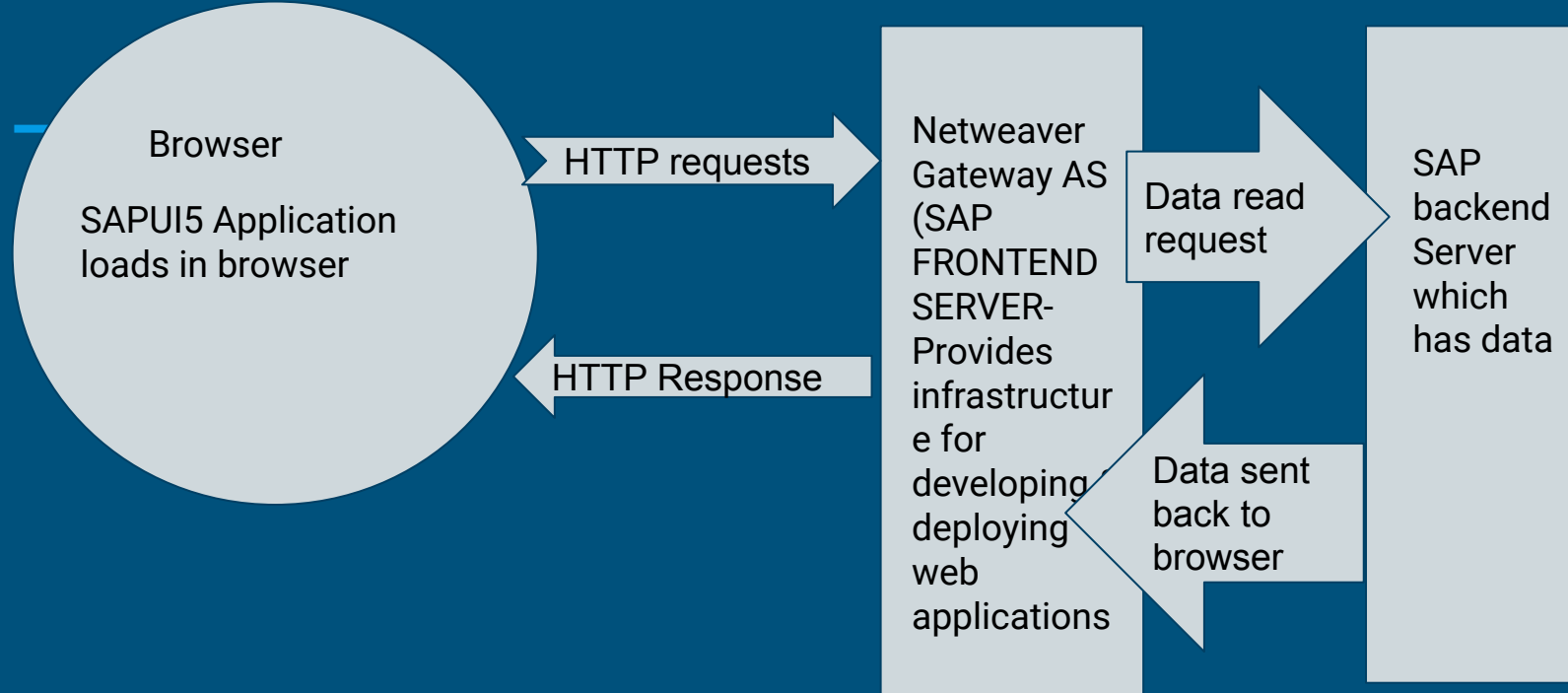
SAPUI5 Applications can run across browsers and devices like tablet, desktop and mobile devices

Data Flow Browser<->Server



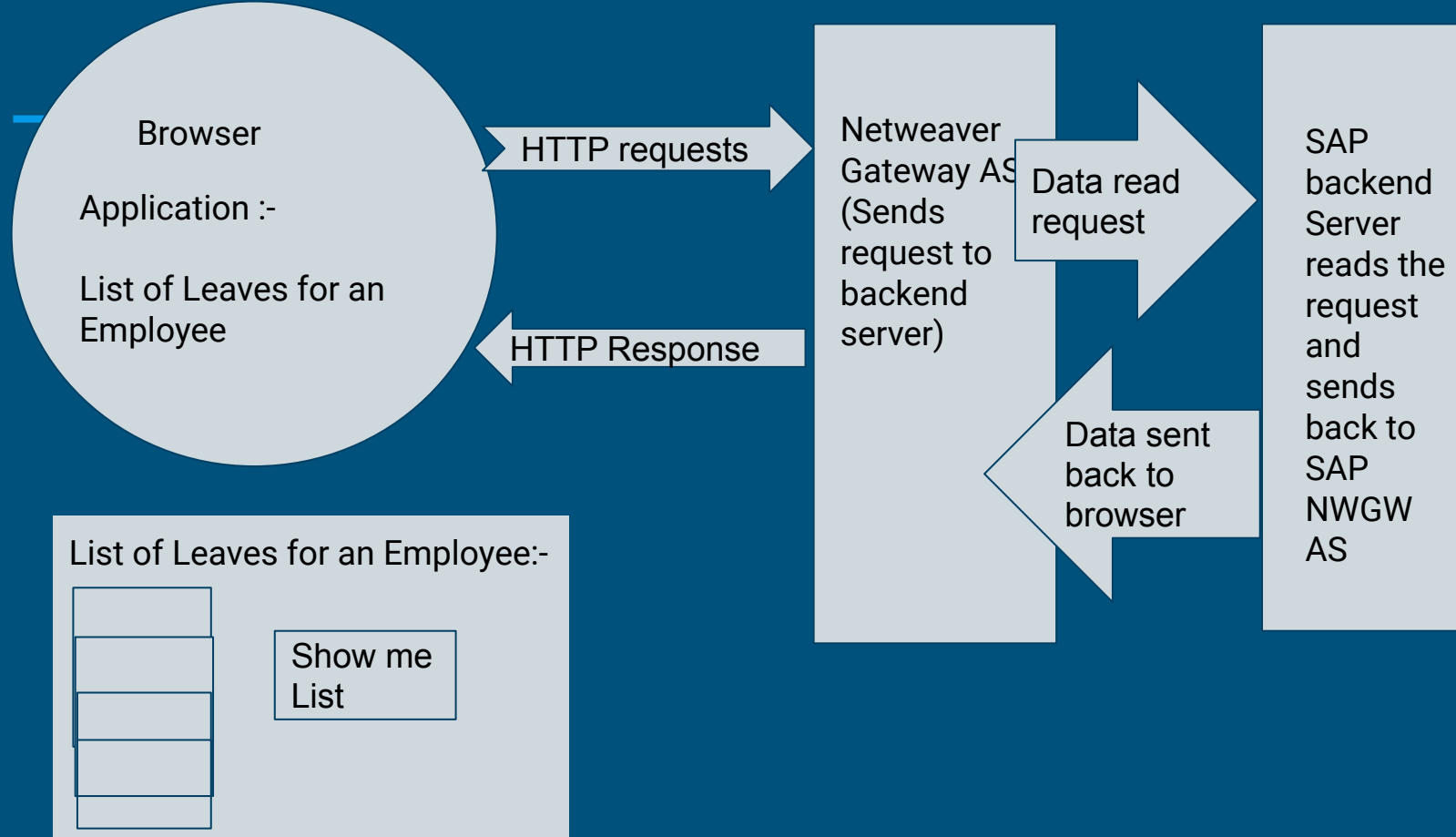
Data Flow Browser<->SAP

Satya Kondiparthi



Data Flow Browser<->SAP

Satya Kondiparthi



SAPUI5 Application is of MVC Pattern:-

- Model, View, Controller

Model:- Stores the data from database or Stores the data entered by User.
For CRUD Operations.

View :- Page which is displayed is View. Presentation logic how view should be displayed.
XML,JS,HTML,JSON

Controller:- Handles **interaction logic by user**. User actions like click of a button are handled by Controller.- JS

Controls :- List, Input fields, buttons , text area etc which are displayed to user in a web page are called Controls. All Controls are placed in a web page .

View and Controller has 1:1 relationship:- Every view has its own controller.

```
{  
  "employees": [  
    {"firstName": "John", "lastName": "Doe"},  
    {"firstName": "Anna", "lastName": "Smith"},  
    {"firstName": "Peter", "lastName": "Jones"}  
  ]  
}
```

JSON stands for JavaScript Object Notation

JSON is a lightweight format for storing and transporting data

XML is a software- and hardware-independent tool for storing and transporting data.

XML is a software- and hardware-independent tool for storing and transporting data.

Satya Kondiparthi

Note

To: Tove

From: Jani

Reminder

Don't forget me this weekend!

```
<note>
```

```
<to>Tove</to>
```

```
<from>Jani</from>
```

```
<heading>Reminder</heading>
```

```
<body>Don't forget me this weekend.</body>
```

```
</note>
```

Set Up programming environment:- SAP Web IDE Personal Edition

- <https://help.sap.com/> - SAP Documentation help
- <https://tools.hana.ondemand.com/#sapui5>
- <http://localhost:8080/webide/index.html>

Prerequisites :- Java Runtime Environment Version 8



Installation issues- webide personal edition

If you are unable to see orion file in your extracted folder , it means its not extracted properly.

1)Please use a software like winzip /winrar/ 7 zip to unzip webide personal edition software.

2)Keep the path of installation as small as possible

3)Downgrade the version of JDK to 1.8 again at the time of unzipping WEBIDE Software. Once its downgraded and once you start using webide personal edition you can upgrade your java version if you want to - This is a last resort to be used

Different Editors which we can use for development of SAPUI5/FIORI Applications:-

- Eclipse(deprecated)

<https://blogs.sap.com/2019/11/26/sapui5-tools-for-eclipse-now-is-the-time-to-look-for-alternatives/>

Very OLD SAPUI5 version is used for development.

- Personal WEBIDE- Only for personal use of educational purposes
- WEBIDE Full Stack(deprecated)- **Still available for the customers who have taken licenses earlier** , even though SAP recommends using BAS(Business Application Studio)

<https://blogs.sap.com/2022/01/10/its-time-to-plan-your-move-to-sap-business-application-studio/>

Modern/current version of SAPUI5 is used

- Business application Studio(BAS)
- Visual Studio Code(VSCode)
- Sublime
- Webstorm

From development perspective of SAPUI5 Applications and usage of SAPUI5 Framework in your respective editors perspective - Any editor can be used, all the editors are same. Where they differ is :- easy of usage, deployment options to backend, deployment time, additional functionality provided in editors like code completion, syntax highlighter, auto correction, additional tools for development, like in BAS we can develop CAPM and RAPM Applications as well along with SAPUI5 Applications, GIT integration - These all fall outside of usage of SAPUI5 framework classes and methods to develop SAPUI5 applications.

Basics of SAPUI5/FIORI concepts are same and development of applications remain same irrespective of development environment, in this course we will try to cover BAS, VSCODE Editor and Personal WEBIDE.

I really hope you found this course valuable,
but either way, please leave a review and
share your experience.

HTML- Hypertext MarkUp Language

- Used to structure web page with HTML Elements
- Complete reference is found at <https://www.w3schools.com/>
- Start tag - <> , End tag - </>

An HTML Element starts with start tag and ends with end tag.

<p>Text section in a paragraph</p>



HTML- Hypertext MarkUp Language

- HTML Document consists

Document type declaration -> `<!DOCTYPE html>`

Header area -> `<head>` \longleftrightarrow `</head>` -- search engine relevant meta information, Link to CSS Files and Javascript files are declared

Document body-> `<body>` \longleftrightarrow `</body>` -- content of web page is declared in body

CSS- Cascading Style Sheets

- To improve look and feel of application we use CSS, by applying them on HTML Elements.
- Syntax :- Selector{

Characteristic: value;

Characteristic: value;

}

CSS- Cascading Style Sheets

- There are three ways of applying CSS
- External CSS File:-
 - `link href="path" rel="stylesheet">`
- At header level of HTML document
 - `<style type="text/css">`

`Selector{`

`Characteristic: value;`

`Characteristic: value;`

`-----`

`}`

`</style>`

- Inline declaration at HTML Element :
`style="characteristic:value"`

Types of selectors in CSS

- Universal
- Type
- ID
- Class
- Descendant
- Group

JavaScript/ECMAScript

- JavaScript is a scripting language to make websites dynamic.

Validate user input, change content dynamically, event handling

- 1995, Brendan Eich, Netscape- Netscape Navigator 2.0
- Microsoft version is JScript.
- Each browser has its own interpretation of JavaScript.
- Standardized version is ECMAScript- 1997.
- Javascript is part of Oracle now .
- SAPUI5 recommends using google Chrome as browser.

JavaScript/ECMAScript

Two ways javascript can be called or included in browser

1) If it is in external file then call it by using:-

```
<script src="path to external JS File"></script>
```

2) If JS Code has to be written in page itself

```
<script> write code between these tags </script>
```

JavaScript/ECMAScript

Two ways javascript can be called or included in browser

1) If it is in external file then call it by using:-

```
<script src="path to external JS File"></script>
```

2) If JS Code has to be written in page itself

```
<script> write code between these tags </script>
```

Basic syntax of JavaScript

- Statements are separated by semicolon
- Single line comment - `//`, multi line comments - `/* Code */`
- Function:- which performs an action.

- Javascript is object oriented language
- Each and every element of a webpage is treated as an object in javascript.
- Window is an object, Console is also an object. We can call window.alert() instead of alert method call directly.
- Window is a special Object hence it can be omitted for alert call.
- JavaScript variables are containers for data values.
- Variables in Javascript are dynamically typed
- Number,string,boolean,Object,Null,Undefined
- Objects are variables too. But objects can contain many values.
- A JavaScript function is a block of code designed to perform a particular task.
- JavaScript methods are actions that can be performed on objects.
- A JavaScript method is a property containing a function definition.
- A JavaScript function is executed when "something" invokes it (calls it).
- Omission of var keyword inside function results in global scope of that variable.
- The JavaScript this keyword refers to the object it belongs to
- JavaScript arrays are used to store multiple values in a single variable
- Comparison operator , difference between == and ===
- Assignment operator '='
- Use Hungarian Notation for declaration of variables.

Advantages of SAPUI5

- As SAPUI5 applications are run in browser - speed of application depends speed of browser.
- As these are web applications - look and feel of applications is very rich

Aggregations

In a composite control, if the relation between parent and child control is such that a child cannot exist without a parent then the relation is called aggregation.

Child control cannot have more than one parent.

Children cannot be placed on their own in a page.

Example:- Radiobutton Group and Radio Buttons

Typed methods-Methods implemented at control level

Inherited methods- Inherited from parent class.

addAggregation(Cardinality 0 to n), setAggregation(Cardinality 0 to 1)

Associations

In a composite control, if the relation between parent and child control is such that a child can exist without a parent then the relation is called association.

Child control can have more than one parent.

Child controls do not share lifecycle of parent.

Example:- Toolbar and Label

Inherited methods- Inherited from parent class.

AddAssociation(Cardinality 0 to n) and setAssociation(Cardinality 0 to 1)

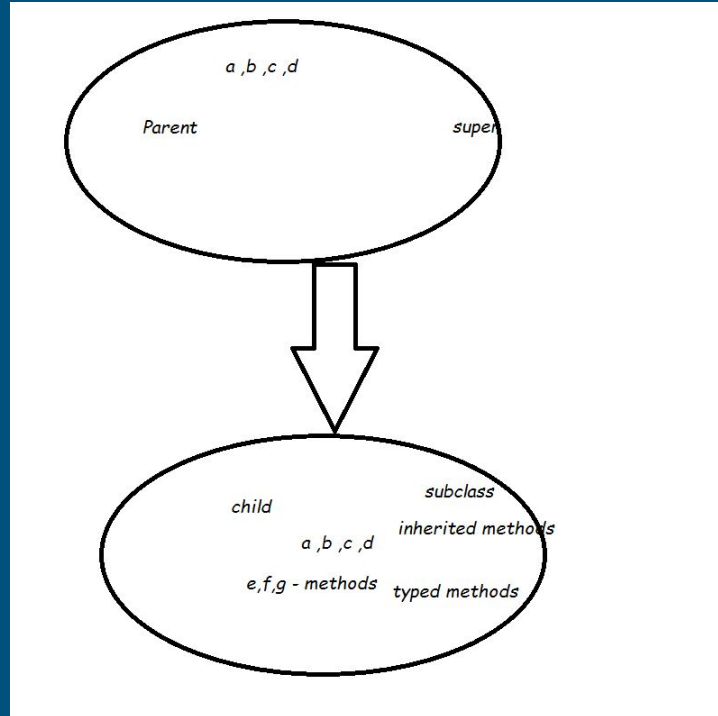
Style class

`addStyleClass` is method on a control to add predefined styles from SDK on any control

eNums or Enumerations

eNums allow developer to define a set of named Constants. While using eNums we can use constants of enum directly in constructor of control.

Use typed methods whenever possible. Use inherited methods only if typed methods are not available for a control.



Layouts:-

Layout controls are used to display elements in a particular manner on screen.

`sap.ui.Layout` - Library

SimpleForm, GridLayout

GridLayout:- GridLayout arranges the controls to fit the size of window when window is resized as per the definition of default span property.

sap.ui.layout.grid



☐ Include deprecated

▼ N sap

▼ N ui

▼ N layout

Grid

GridData

GridIndent

GridPosition

GridSpan

class sap.ui.layout.Grid

Overview

Constructor

Properties

Aggregations

Associations

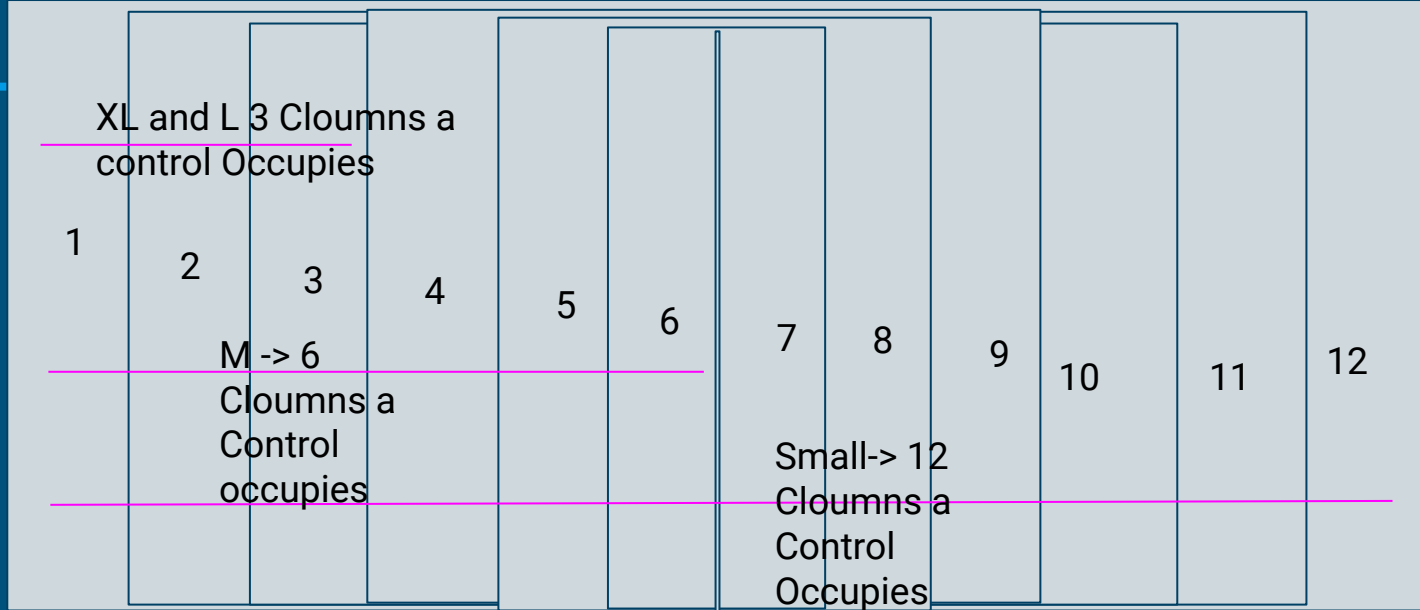
Events

Methods

Name	Type	Default Value	Description
containerQuery	boolean	false	If set to true, the current range (large, medium or small) is defined by the Visibility: public
defaultIndent	sap.ui.layout.GridIndent	XL0 L0 M0 S0	Optional. Defines default for the whole Grid numbers of empty columns b Allowed values are separated by space Letters L, M or S followed by num or 14 m4. Note: The parameters must be provided in the order . Visibility: public
defaultSpan	sap.ui.layout.GridSpan	XL3 L3 M6 S12	Optional. A string type that represents the span values of the Grid for larg followed by number of columns from 1 to 12 that the container has to tak Note: The parameters must be provided in the order . Visibility: public
hSpacing	float	1	Optional. Defines the horizontal spacing between the content in the Grid Visibility: public

As per defaultSpan property of sap.ui.Layout.Grid a control occupies columns as below, as per different screen sizes.

Satya Kondiparthi



As per defaultSpan property of sap.ui.Layout.Grid a control occupies columns as below, as per different screen sizes.

Satya Kondiparthi

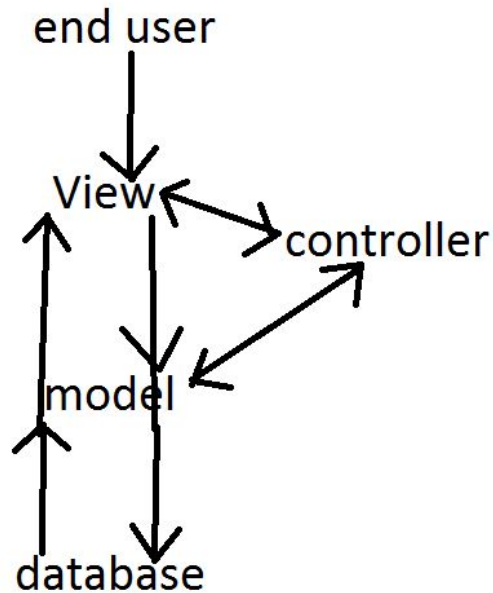
Small screens -occupying 12 columns in SAPUI5 application in **GRID Layout**

Medium screens-occupying 6 columns in app

XL,LARGE(L)
screens-3 columns

1	2	3	4	5	6	7	8	9	10	11	12
---	---	---	---	---	---	---	---	---	----	----	----

Model View Controller



view displays data- contains controls in which data is displayed

model holds the data from database and can be used to set the data back to database or display it on the view

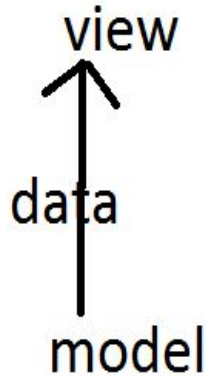
controller code can be triggered for any user interaction

- **There can be N number of models in an application**, one model can hold translatable texts, another model can hold data of Products, another model can hold contact persons data etc
- **An application can have any number of views** , for example one view can display detailed information and another view can display generic information
- **View can contain another view**
- Each and every view can have a controller or all views can have a single controller. Controller handles user actions of the views.
- **It is possible that view does not have any controller.**

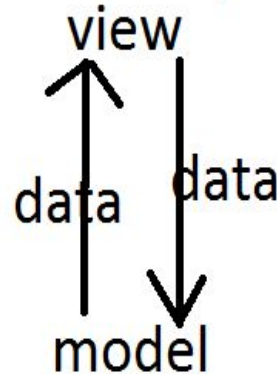
JSON MODEL:-

- `sap.ui.model.json.JSONModel` is the API for handling data in the model
- Two way data binding
- Client side model

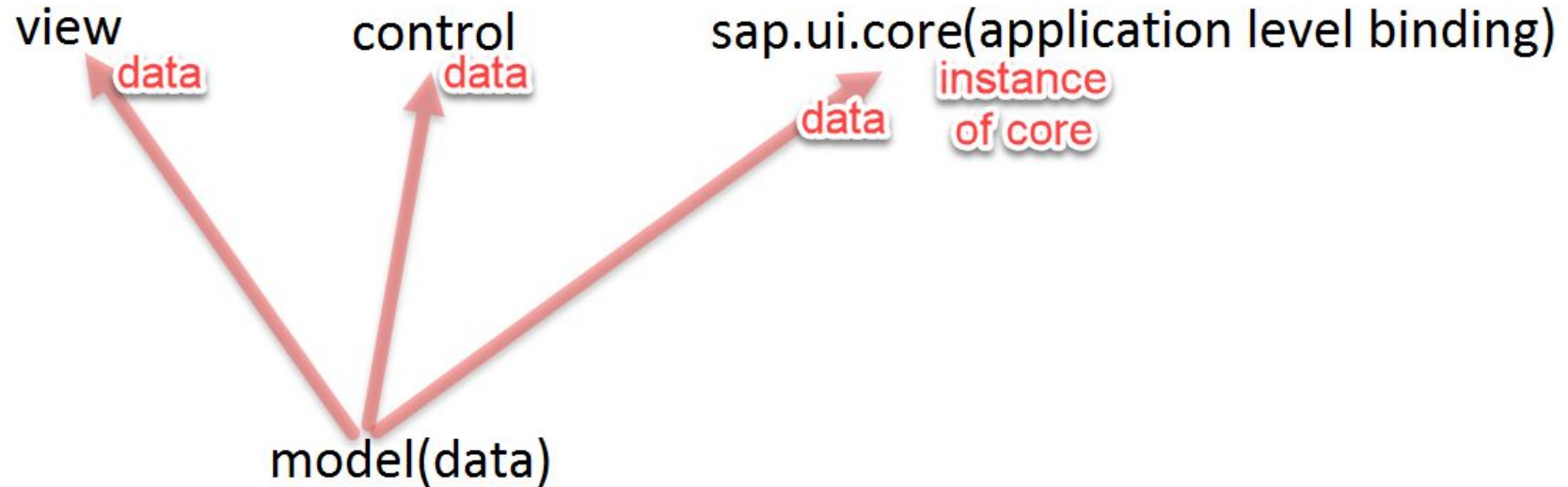
one way data binding



two way data binding



Binding the data to appear in the view :-if we bind data to instance of sap.ui.core then it can be accessed anywhere in application



Absolute binding Path and relative binding path:-

Satya Kondiparthu

/SweetsSupplier/1/Address/Street to Address Street Property in property binding-2nd array element

```
var oData={
  "CountSweets": "2",
  "SweetsSupplier": [
    {
      "ID": 0,
      "Name": "Sweet Magic",
      "Address": {
        "Street": "Sivarao Street",
        "City": "Vijayawada",
        "State": "Andhra Pradesh",
        "ZipCode": "521456",
        "Country": "INDIA"
      }
    },
    {
      "ID": "1",
      "Name": "Aanjaneya Sweets",
      "Address": {
        "Street": "Bhavanipuram",
        "City": "Vijayawada",
        "State": "Andhra Pradesh",
        "ZipCode": "521456",
        "Country": "INDIA"
      }
    }
  ]
};
```

**/SweetSupplier is
bound to table-
Absolute Path - starts
with /**

**Relative Path- Table
Columns are bound-
{Name} and {ID}**

- **Simple Binding** : Property Binding

Binding Property of a control to property of Model ,ex:- text : "ID"

- **Complex Binding** : Property Combined with a text

Ex:- text : "Number of sweet Suppliers: {/SweetsSupplier/1/ID}"

- **Aggregation Binding** : Binding happens on an aggregation of a control . In general to Array of elements in model.

Ex:-

```
oTable.bindItems({  
    path : "/SweetsSupplier",  
    template : oTemplate  
});
```

View types:

- XML-recommended by SAP/commonly used
- JavaScript- after XML, this is most preferred type
- JSON
- HTML

XML,JSON,HTML views are Declarative view types.

Naming convention for creating views:-

XML View:-	ViewName.view.xml
JS View:-	ViewName.view.js
JSON View:-	ViewName.view.json
HTML View:-	ViewName.view.html

Controller naming convention:-

ControllerName.controller.js

MVC pattern:-

- ControllerName should be same as ViewName
- Views should be created under view folder
- Controllers should be created under controller folder

Factory Function:- A function which returns an object without using new keyword

Namespace : its an unique identifier for application, its mandatory when app is run inside FIORI Launchpad.

We also need to specify location of files with respect to namespace, which is done using

data-sap-ui-resourceroots parameter of script tag. ./ -current directory

or using **jQuery.sap.registerModulePath('satya.prasad.mvcapp', './webapp/');**

Whenever framework comes across namespace , it will try to find the resources specified in the path in parameter of data-sap-ui-resourceroots in bootstrapping

Views Controllers, controls etc can be considered **modules**. These modules are loaded on demand separately for improvement of performance.

Asynchronous module definition(AMD):- (Found in SAP.UI Namespace)

- To arrange the code in modules,
- load the modules asynchronously
- To resolve dependency issues

To define the modules and load dependencies asynchronously we use **sap.ui.define()**.
To just load dependencies we use **sap.ui.require()**- this is not used to define modules.

Synchronous module definition and loading will be taken care by **jQuery.sap.declare** and **jQuery.sap.require** classes (does not create module but just loads them synchronously).

We use **Factory Functions** for creating view:-

`sap.ui.view()` or `sap.ui.jsview()` - for creating JS View

Note:- Function which returns object without using NEW Keyword is called a factory function.

Advantages of using XML Views:-

- 1) Clear separation of view and controller logic.
- 2) Layout editor can be used to design XML Views
- 3) All Fiori applications are developed using XML Views
- 4) Structure of controls we place in XML Views match the structure of UI5 Application, it is not the case with Javascript views.
- 5) Automatic validation happens in XML Views, when we use SAP WebIDE

Component:-

- Component is a javascript file in SAPUI5 application , which is started first when the application starts from FIORI Launchpad
- Component defines service URL, instantiates models, creates Initial view, sets up routing.
- RootView can be created with below metadata section of component

```
metadata:{  
    "rootView":"satya.prasad.mvcapp.view.App",  
    "Config": {  
        "serviceUrl":"path to JSON File"  
    }  
},
```

Even if our createContent custom implementation is not there, rootView is still created based on prototype's createContent method.

Routing:-

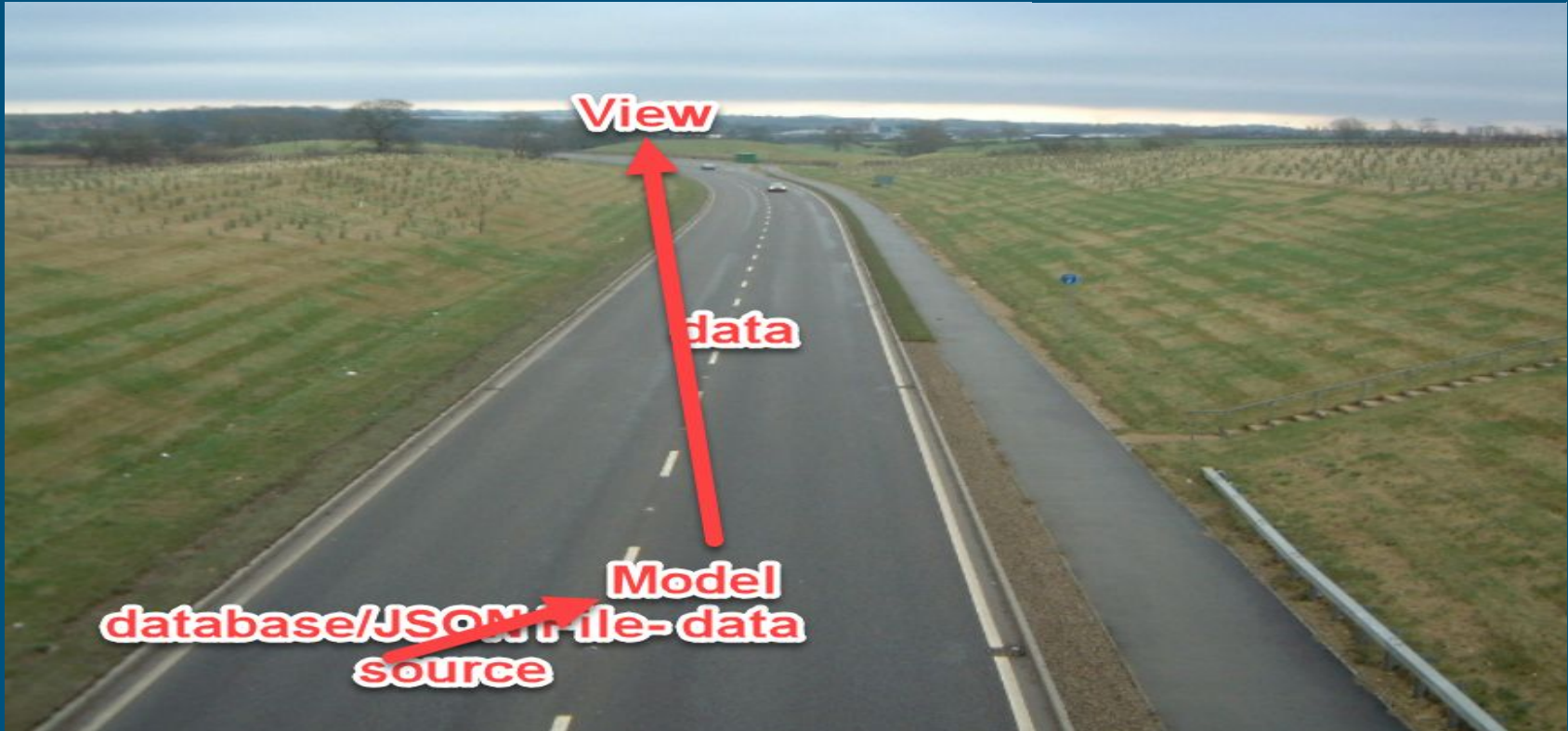
- Navigation
- Bookmarking deep URLs and share them
- Refresh of the page will not direct us back to home page

Routing works based on URL Patterns(# based navigation) of SAPUI5 application:-

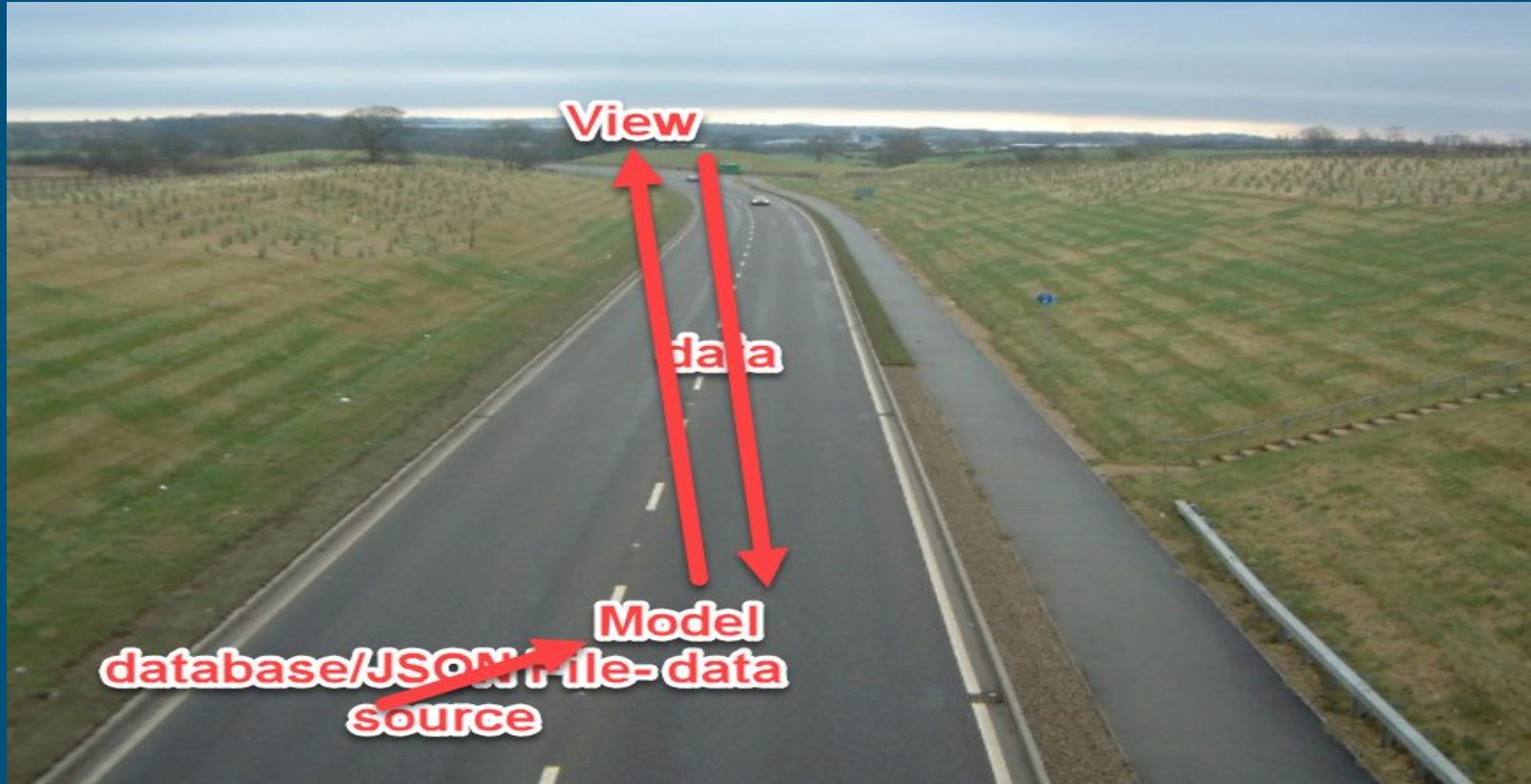
- index.html# -> UI5 application launches starting page
- index.html#/detail/0 -> UI5 application directly launches second.view.xml to see detail of sweet shop ID

One way data binding : Data binding is from model to the view, whatever data is there in model is reflected in the view, but data changed in view will not be reflected back into model

Satya Kondiparthi

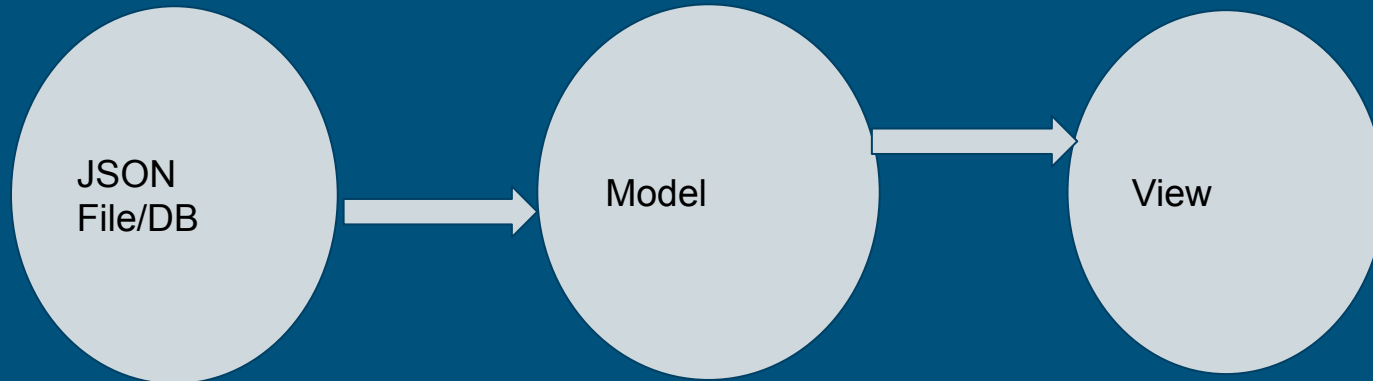


Two way data binding : Data binding is from model to the view and again back to the model as well. Data in both view and model are always in sync
Satya Kondiparthi

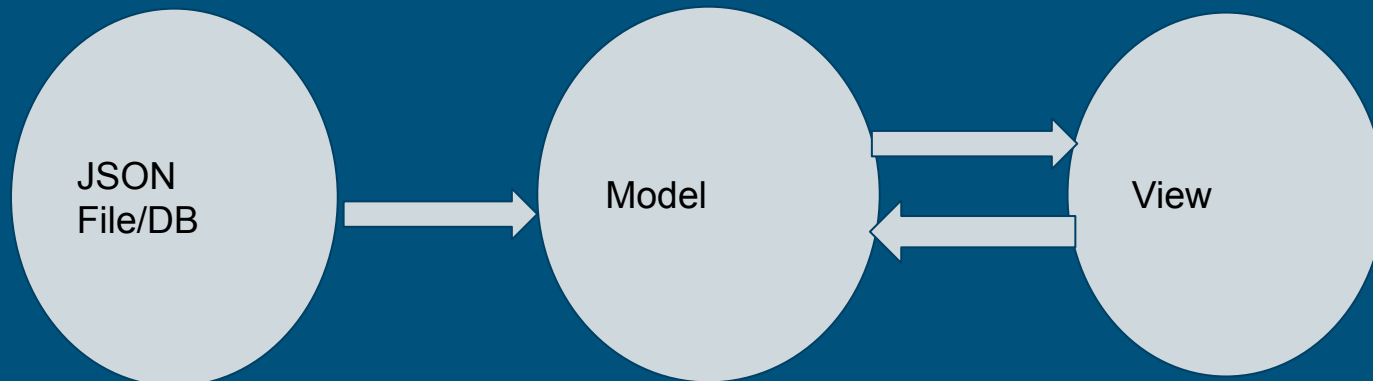


One way data binding

Satya Kondiparthi



Two way data binding



One Way binding Example:-OData Model binding Satya Kondiparthi

Two way data binding Example:-JSON Model binding, XML Model binding

oModel.getProperty("binding path");

oModel.setProperty("binding path", value);

oModel.setDefaultBindingMode("OneWay");

oModel.setDefaultBindingMode("TwoWay");

Binding Types:-

Satya Kondiparthi

1)Property binding: Controls property is bound to model

2)Expression binding: Controls property bound to expression

Two way binding stops working when we use expression binding

Similar to Conditional Operator in Javascript:-

```
condition ? exprIfTrue : exprIfFalse
```

3)Aggregation binding: Control's aggregation property- bound to array path, in model

4)Element Binding:- Every element/item of an array in model is called an element, if that element is bound to property of a control , then that binding is called element binding.

Connecting to backend SAP System:-

<https://www.sap.com/index.html>

Register with a valid email ID in the site above.

<https://register.sapdevcenter.com/SUPSignForms>

Register in Gateway Demo system

Change initial password generated with below URL

<https://sapes5.sapdevcenter.com/sap/bc/gui/sap/its/webgui>

This is SAP System provided by SAP Company for educational purposes .SAP is sole owner of this system and all Copyrights are reserved with SAP .

[https://sapes5.sapdevcenter.com/sap/opu/odata/iwbep/GWSAMPLE_BASIC/?\\$metadata&sap-ds-debug=true](https://sapes5.sapdevcenter.com/sap/opu/odata/iwbep/GWSAMPLE_BASIC/?$metadata&sap-ds-debug=true)

https://sapes5.sapdevcenter.com/sap/opu/odata/iwbep/GWSAMPLE_BASIC/ProductSet?sap-ds-debug=true

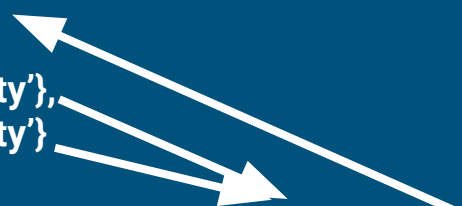
Formatter:-

- Instead of displaying the data(model property value) as it is from backend we can modify data to be user friendly/understandable format(custom value different from backend).
- Instead of binding the data as is from backend , we can bind different value from backend to a control.
- Expression binding is used to evaluate simple true/false conditions , but formatter can be used to evaluate more complex conditions and pass back the value to be bound to a property.

Syntax used:-

Binding with multiple parts in View

```
ControlProperty="{
  parts: [
    { path:'modelProperty'},
    { path:'modelProperty'}
  ],
  formatter: '.controllerProperty.formatterFunction'
}"
```



The diagram consists of two white arrows on a dark blue background. One arrow originates from the text 'Binding with multiple parts' and points to the 'parts' array in the JSON snippet. The second arrow originates from the same text and points to the 'formatter' property in the same snippet.

Require modules in XML Views:-

- We can use these modules as formatters, event Handlers etc
- We can avoid global variables, we do not have to interact with Controller for using helper functions

```
• core:require="{  
  Box: 'sap/m/MessageBox',  
  Toast: 'sap/m/MessageToast'  
}"
```

```
<Button text="Press Me!" press="Box.show('Hello!')"/>
```

```
• xmlns:core="sap.ui.core" core:require="{Util:'satya/prasad/mvcapp/utilities/utilities'}">
```

```
<ObjectNumber number="{ODATA>WeightMeasure}" unit="{ODATA>WeightUnit}"  
  state="{  
    parts:[  
{path:'ODATA>WeightMeasure'},{path:'ODATA>WeightUnit'} ],  
    formatter:'Util.formatterFunction'  
  }"></ObjectNumber>
```

- `core:require="{formatMessage:'sap/base/strings/formatMessage'}">
<Label text="{ parts:['i18n>Name','Name'], formatter:'formatMessage'}"></Label>`

Using Standard Data Types for formatting and validation:-

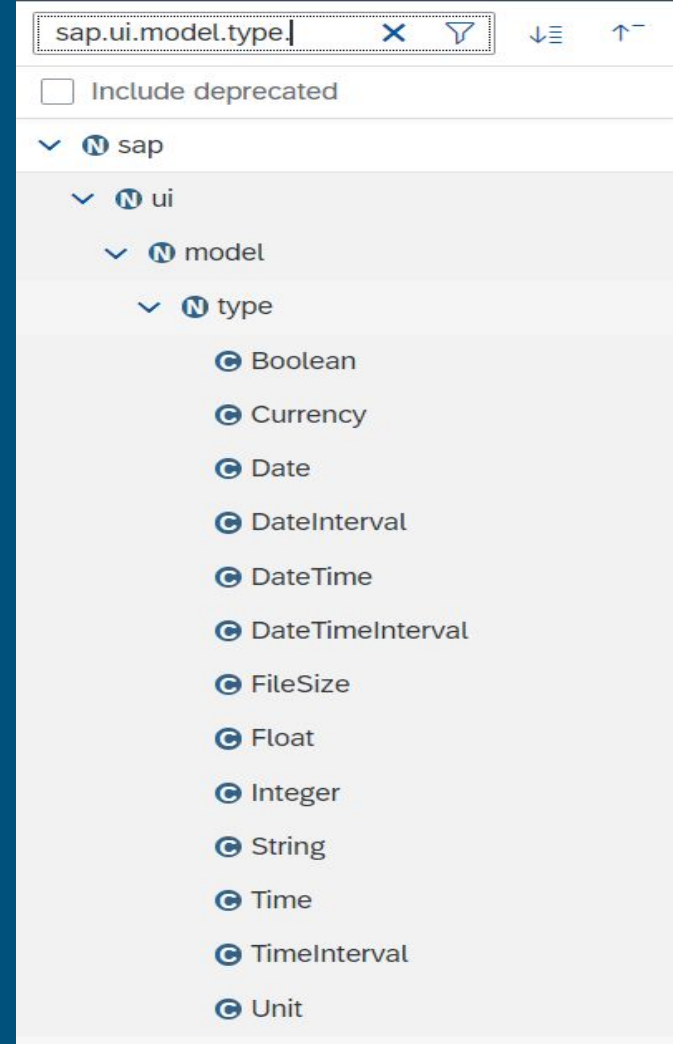
Difference between using formatter and using data types:-

Formatter- Model data is not touched, but UI display value is modified as per implemented formatter function- **Two way data binding do not work in formatter**

Formatter is used only for read only fields

Data Type Usage:- UI Value is modified as per binded data type for display purposes, but also as per user input - model data is also parsed back to its original data format

Data type is used for input fields



Element Binding:- When an individual element of an array in model data is bound to a control - data in model is displayed in control.

```
{
  "CountSweets": "2",
  "SweetsSupplier": [{
    "ID": 0,
    "Name": "Sweet Magic",
    "Address": {
      "Street": "Sivarao Street",
      "City": "Vijayawada",
      "State": "Andhra Pradesh",
      "ZipCode": "521456",
      "Country": "INDIA"
    }
  }, {
    "ID": "1",
    "Name": "Aanjaneya Sweets",
    "Address": {
      "Street": "Bhavanipuram",
      "City": "Vijayawada",
      "State": "Andhra Pradesh",
      "ZipCode": "521456",
      "Country": "INDIA"
    }
  }
}]
```

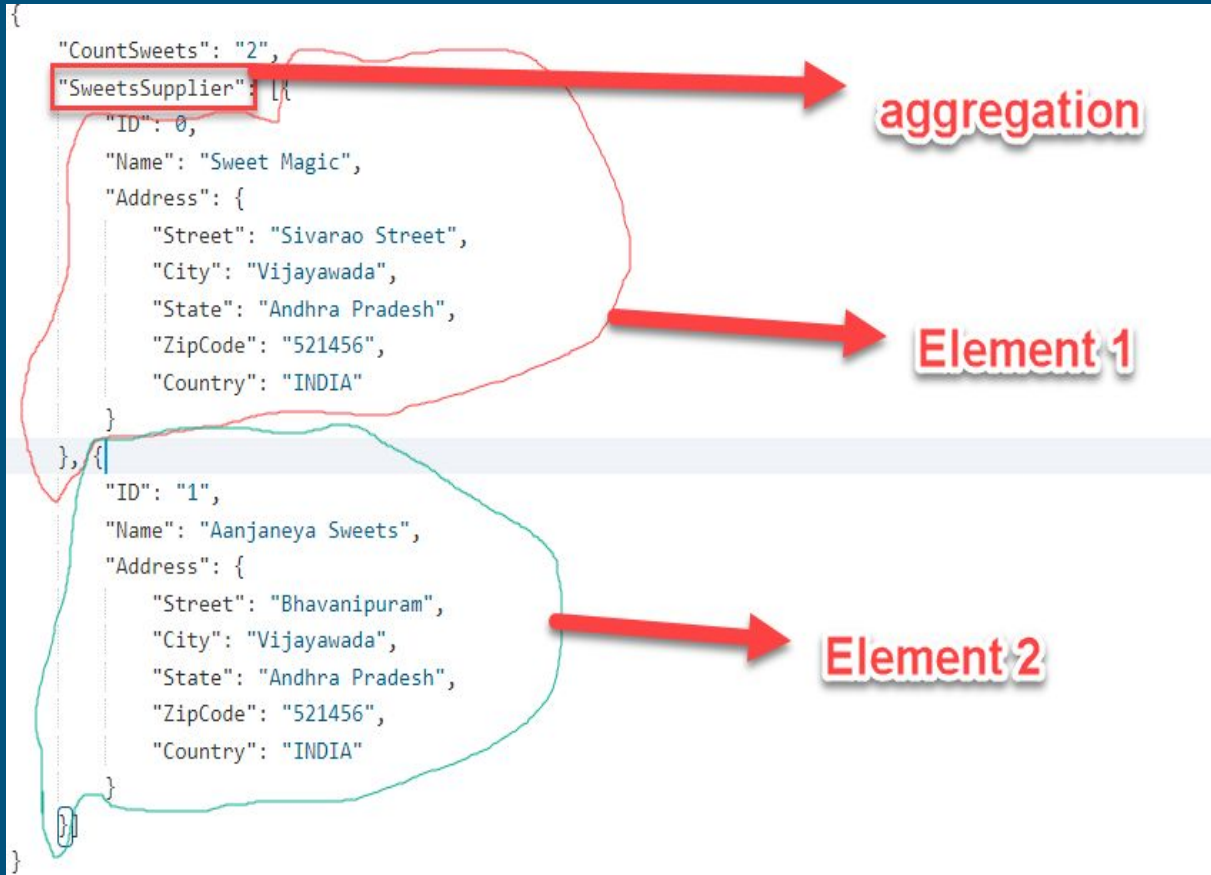
**Elements of
SweetsSupplier
Object**



/SweetsSupplier/0 and
/SweetsSupplier/1 are
paths associated with two
elements shown here for
SweetsSupplier array.

**control.bindElement(path to
element)** ->method used to
do element binding

Difference between Aggregation binding and Element binding





If aggregation/array is bound to a control then its called aggregation binding.

If Element of an aggregation/array is bound to a control then it is called element binding

Description=ES5
Type=HTTP
TrustAll=true
Authentication=BasicAuthentication
Name=ES5
ProxyType=Internet
URL=https://sapes5.sapdevcenter.com
ProxyType=Internet
WebIDEUsage=odata_abap,dev_abap,odata_gen,ui5_execute_abap,bsp_execute_abap,odata_xs
WebIDESystem=ES5
WebIDEEnabled=true
sap-client=002

E:\web ide\eclipse\config_master\service.destinations\destinations

Name	Date modified	Type	Size
 ES5	7/18/2021 6:54 PM	File	1 KB
 S4H	12/3/2020 4:39 AM	File	1 KB

Enhancing standard Fiori Applications:-

- 1)View Modification ->hide or show elements
- 2)View Extension-> in between View extension and view Replacement we need to go with View extension
- 3)View Replacement->if there is no view replacement option available then use this option
- 4)Controller Extension-first preference should be for controller replacement not for controller extension
- 5)Controller Replacement

We should always make changes in extension project only, never in original project

Master Page S2

Detail Page S3

Further detail Pages S4,S5 etc...

For view extension, go to Configuration.js file and find odata URL and go to source code and write

SAPUI5/FIORI Development with VSCode

- All core SAPUI5/FIORI Concepts remain same , whether we use VSCode or BAS or Personal WEBIDE or WEBIDE Full Stack or SUBLIME OR WEBSTORM.
- Configuration files when connecting to SAP backend, Code correction, Code completion, Code beautification, Editor themes, support for ODATA V4 versions, the way we deploy applications to backend or to cloud environments, code minification options etc may vary when/while using newly introduced Editors.
- Newly introduced technologies/tools/frameworks like CAPM,RAPM,AI Tools, Low Code/No Code tools may not be supported by older editors and newer editors may support all these technologies/frameworks/tools

SAPUI5/FIORI Development with VSCode

- Disadvantage of BAS for students(not for SAP Customers) is after every 3 months BTP trial expires , now , as per current policies of SAP in June 2023, these policies change over from time to time. VScode do not have this problem , we can use it until SAP Supports usage of VSCode.
- BAS do not need any installation as its available on the cloud, but VSCode needs to be installed and system should meet minimum requirements specified in documentation of VSCode.
- BAS is copied or created using VSCode, BAS is optimized to be used by SAP for its development environment, but VSCode is most used editor in the world and can be used to develop variety of applications like react, angular,C++ etc..

VSCode installation:-

1) Install node js

Confirm installation is successful in CMD:- `node -v` or `npm -v`

2) Install SAPUI5 tooling :-

`npm install -g @ui5/cli` or `npm i -g @ui5/cli`

Confirm installation with command :- `ui5 --help`

`npm i -g @sap/ux-ui5-tooling`

Confirm installation with command :- `fiori --help`

3) Install VSCode

Node.js

- Node.js is a runtime environment which developers need to run javascript on the server
- Chrome provides its own runtime environment to run Javascript in browser- to handle click events, to navigate to other pages etc, it provides its own functions & Objects.
- Node.js provides its own runtime environment to run javascript in server- like libraries to setup web server or libraries to integrate with File System.

Chrome Browser

Javascript Engine V8 +
Additional functions provided by
Chrome to run in browser

Node JS

Javascript Engine V8 +
Additional functions
provided by Node to run in
Server

V8 is Google's Open Source project-> Same V8 engine is used to create NodeJS

- We can build web servers, send emails, access File System on computer to read and write Files etc.. using Node JS

Javascript---> V8 Engine→Machine Code→ Execute program on server

Chrome:-
Normal Standard JS + additional Objects
like window, document

Node:-
Normal Standard JS + additional objects like
global, process

Synchronous function calls/ Blocking I/O

A Function

B Function

C Function

E
x
e
c
u
t
i
o
n

$A + B + C = \text{Total execution time}$

Asynchronous function calls/ non Blocking I/O

A Function

B Function

C Function

E
x
e
c
u
t
i
o
n

Either of A or B or C- longest function call to execute=> Total time of execution

- Node.js uses a non-blocking I/O-asynchronous functions calls to perform operations- Eg like reading files on the system or fetching data from database, just like a browser.
- NPM is largest ecosystem in the world-<https://npmjs.com> , NPM packages are predefined packages for writing server side code.
- Write First Node.js Program:- Javascript program in Node.js environment.

Loading Node Modules

- Some modules are globally available like console.log we don't need to load them separately.
- Loading Core Node Modules- These modules are included with Node installation . Eg FileSystem module
- Load our own modules which we created: 1) module.exports , 2) require
- Load Third Party modules- Modules written by other developers- When we installed NPM a linkage to npmjs.com is created from where we can load NPM Modules into our application.

a)Initialize NPM init- this will create package.json file in our Application/Project folder.While creating Package.json file systems prompts us to enter details, we just ignore them and leave them to default values- as those are relevant for package creation, in our case we are trying to use package.

b)npm install <packagename>(@VersionNumber)- this will update package.json, create node_modules folder with all downloaded code, create package-lock.json file.

UI5 Tooling for testing applications

- ui5/cli,@sap/ux-ui5-tooling these are two NPM modules we have already installed globally, we can find relevant documentation and commands in NPMJS.COM website, one of the uses of ui5/cli is to start a web server to **test SAPUI5 applications in VSCODE**.
- PREREQUISITE:- Just like any other NPM module to be used, here as well we need to provide a linkage for application to NPM website with 'NPM init' command, then 'ui5 init'(which will create ui5.yaml file) and 'ui5 serve' to create a web server to test SAPUI5 applications.

NOTE:- Before creating web server (i.e. using ui5 init or ui5 serve) make sure manifest.json file exists in our application. Also all files of application should exist inside webapp folder.

```
{  
  "sap.app": {  
    "id": "satya.prasad.mvcapp",  
    "type": "application"  
  }  
}
```