|  |  |
| --- | --- |
|  |  |
| Weekly Portfolio |  |
|  |  |
|  | Responsive Web Applications |
|  | By Ashan HewagamaBSCP|CS|51|087 My GitHub link - [My Github Repository](https://github.com/ifunboi/-SIT120-RWA-Assignments.git) |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | ContentWeek 1………………………………………………………………….Page 3-4  * Responsive web apps * HTML * CSS  Week 2………………………………………………………………….Page 6-8  * Javascript  Week 3………………………………………………………………….Page 9-16  * Vue Framework  Week 4………………………………………………………………….Page 18-24  * User Input Handling with Vue  Week 5………………………………………………………………….Page 24-25  * Computed properties  Week 6………………………………………………………………….Page 26-27  * Vue Components  Week 7………………………………………………………………….Page 27-28  * Vue Components – Props  Week 8………………………………………………………………….Page 28  * Component Events | |  |
|  |  |  |  |

**Week 1 summary**

So in week one we have learnd what and how responsive web applications work and how to create our own web apps using Html(Hypertext Markup language) and CSS(Cascading Style Sheets) as a start.

Tags in HTML

So the tages are like keywords which defins how the web browser will format and display the content, and with the help of html tags the web browers can say which is html and which is normal content.

HTML tags has 3 main parts, <opening tag > --- content----- </closing tag> but in **some cases there are no closing tags.**

* All HTML tags must enclosed within < > these brackets.
* Every tag in HTML perform different tasks.
* If you have used an open tag <tag>, then you must use a close tag </tag> (except some tags)

So when coding with html first we have to make the main tag which is <html> and then the<head> after that the <body> in the body is where we put all our content.



Doctype

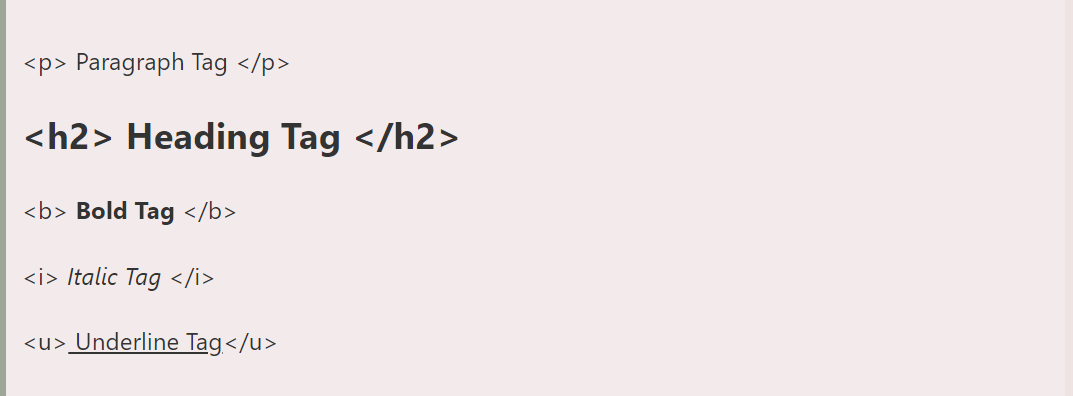
● The <!DOCTYPE> declaration must be the very first thing in your HTML

document, before the <html> tag.

● The <!DOCTYPE> declaration is not an HTML tag; it is an instruction to the

web browser about what version of HTML the page is written in.

**so lests talk about some basic tags**, or text formating tags…



So the the <h1> tag is for the heading ,and there are six sizes <h1> being the biggest and <h6> being the samllest.

**Comments** – to add an comment in html we use this <!-- The comment here -->

**Inserting images** – to insert an image we use the tag <img> and unlike other tags this tag does not hav an close tag ,

* Type <img src = “image.ext”>, where image.ext indicates the location of the image file
* The width=n and height=n attributes can be used to adjust the size of animage+



**Altrnate Text**

* Some browsers don’t support images. In this case, the ALT attribute can be

used to create text that appears instead of the image.

Example:

<img src=“dog.jpg” alt = “Picture of a Dog”>

**Links**

• A link lets you move from one page to another, play movies and sound, send

email, download files, and more....

• A link has three parts: a destination, a label, and a target

• To create a link type

<a href=“page.html”> label </a>

****

CSS ( Cascading Style Sheets )

Cascading Style Sheet, or CSS. A text file with an extension is a typical CSS file. css and consists of a number of instructions or rules. These guidelines specify how to show in HTML.

HTML and CSS work together to produce beautiful and functional Web sites

● HTML = structure

● CSS = style

There are 3 ways to attach CSS to a page:

1. External Style Sheet: Best used to control styling on multiple pages.

<link rel="stylesheet" type="text/css" media="all" href="css/styles.css" />

2. Internal Style Sheet: Best used to control styling on one page.

<style type=“text/css”> h1 {color: red}

</style> Attaching a Style Sheet

3. Inline Style Sheet\*: CSS is not attached in the <header> but is used directly

within HTML tags.

<p style=“color:red”>

Some Text</p>

**Week 2 Summary**

1. **Javascript**

So in week 2 we talk about **Javascript,** so Javascript is an text-based programig language used on both client-side and sever-side to make our web pages more interractive. While HTML and CSS give Sturcture and style to web pages , Javascript gives the interractve elements that engage an user. Nowadays the use of JS is so high almost every web page in the internet has JS elemnts embeded in them.

Three ways of using JavaScript in a web page :

**External JavaScript –**

* External JavaScripts are scripts that are written and saved in a separate document



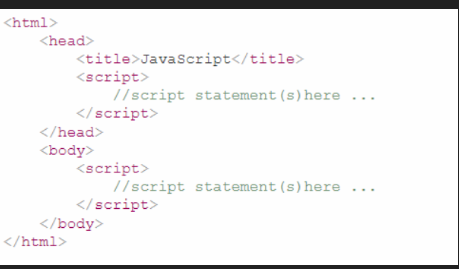
**Internal JavaScript –**

* Internal JavaScripts are scripts enclosed between the tag <script> and <script> and
* embedded within the head or body section of the Web documents



**Inline JavaScript –**

* Included within the values of attributes (Event Handler)

******

**There are 3 kinds of declarations in JS –**

Var – this declares loacl or and globle variable

Let --- declaers a local variable , optionally initialise it to a value

Const--- declaers read-only named constant

**Comments—**

Single line comments starts with --- // (duble slash)

Multiline comments starts with -- /\* the comment

The comment \*/ (embedded within)

**Advantage of JavaScript function**

**1.Code reusability:**

We can call a function several times so it

save coding.

**2.Less coding:**

It makes our program compact. We don’t need to

write many lines of code each time to perform a common

task.



*In this example, the JavaScript code, function myFunction ( ), is in the <head> section of the page.However, you can also put it into the <body> section, the effect will be the same.*

**Week 3 summary**

Vue FrameWork

* So the Vue framework is used create user interfaces. And it builds on top of standard HTML, CSS and of course Javascript.
* The mix of all these provides a declarative and component-based programming model that helps you efficiently develop user interfaces, be it simple or complex.

Core Features of Vue are 🡪

1.Declarative Rendering

Vue extends standard HTML with a template syntax

that allows us to declaratively describe HTML output based on JavaScript

state.

2.Reactivity

Vue automatically tracks JavaScript state changes and efficiently

updates the DOM when changes happen.

**The Progressive Framework**

• Vue is designed to be flexible and incrementally adoptable

• Depending on your use case, Vue can be used in different ways:

• Enhancing static HTML without a build step

• Embedding as Web Components on any page

• Single-Page Application (SPA)

• Fullstack / Server-Side Rendering (SSR)

• Jamstack / Static Site Generation (SSG)

• Targeting desktop, mobile, WebGL, and even the terminal

**Vue Instance**

• Every Vue application starts by creating a new Vue instance with

the Vue function

Var vm = new Vue ({

//Options

})

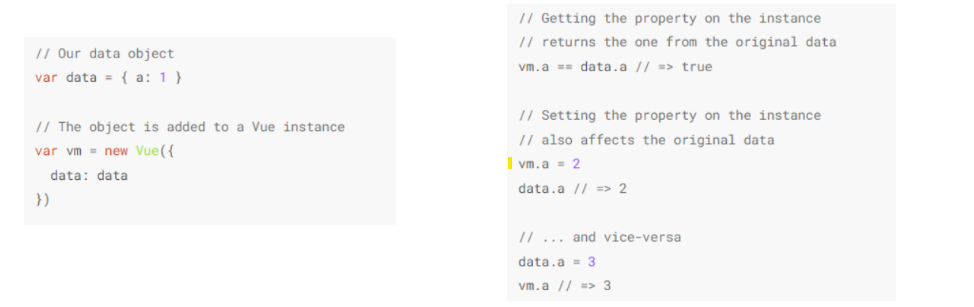
**Data and Methods**

• When a Vue instance is created, it adds all the properties found in

its data object to Vue’s reactivity system

• When the values of those properties change, the view will “react”,

updating to match the new values.

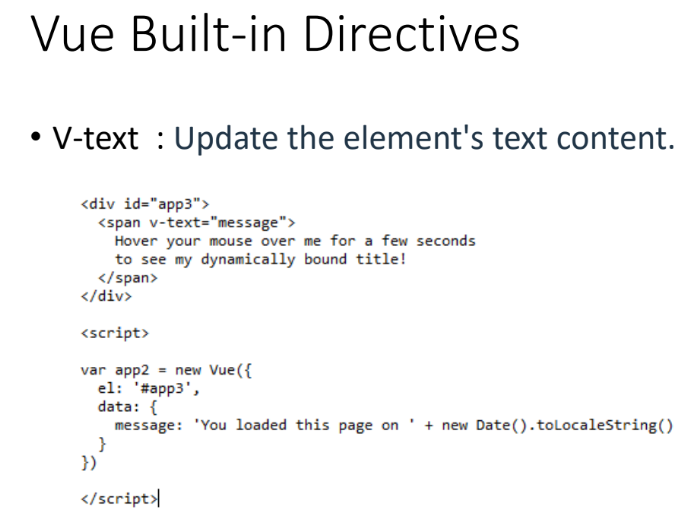
******

Lets look at some Examples ……

***Exsample 1***

******

******

******

******

**Conditional Rendering**

• To show content based on certain conditions

• When building your Vue.js application, there is usually a need to

display template elements based on things like data, user behavior or

input.

Ex: based on user type (regular or admin) we will show/hide a button,

menus or others part of our page

**Types**

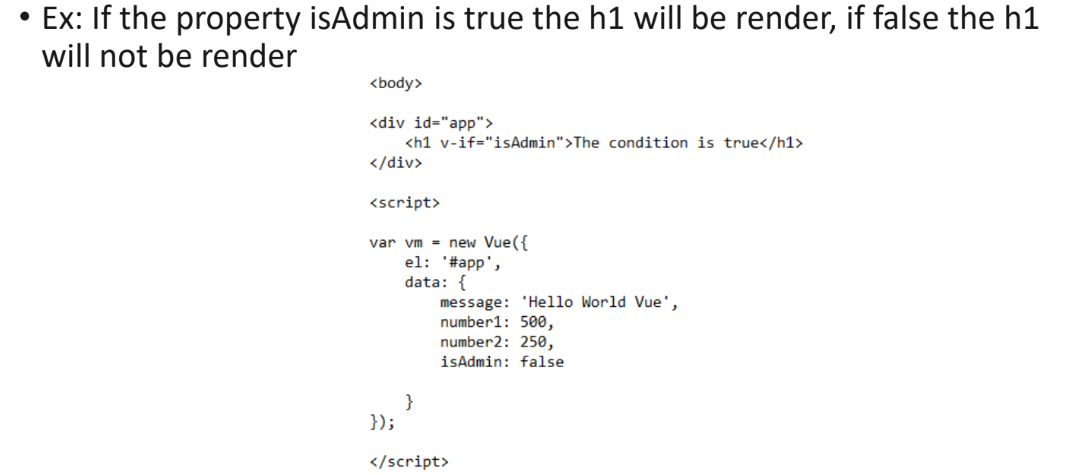
• Vue has a few directives that you can use to achieve conditional

rendering in your application easily.

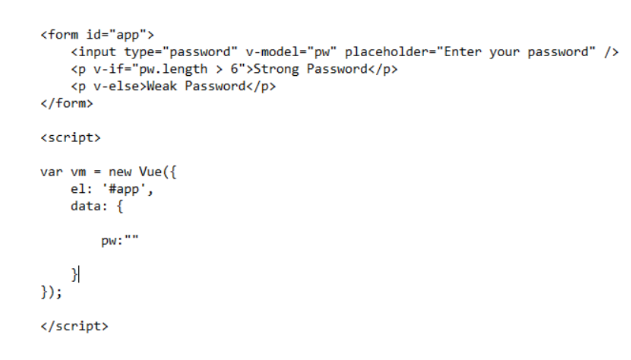
* 1. The **v-if** directive

• The v-if directive will display any template element it is attached to as

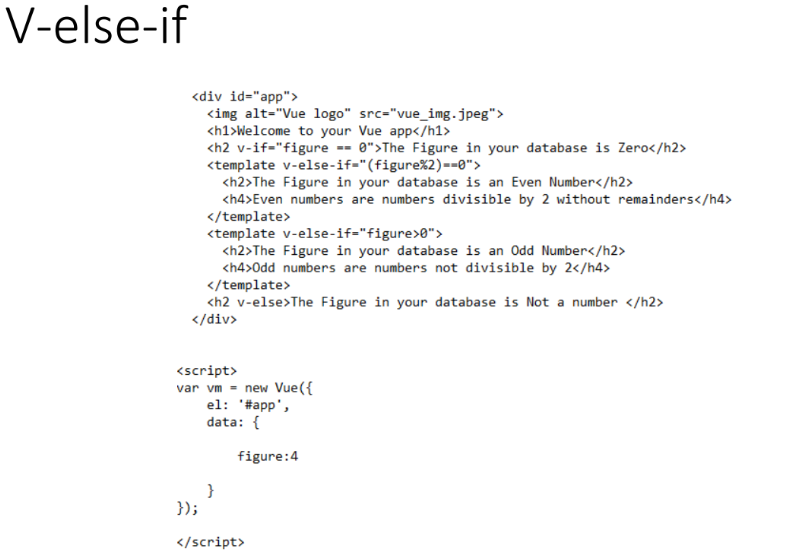
long as the condition set is met.

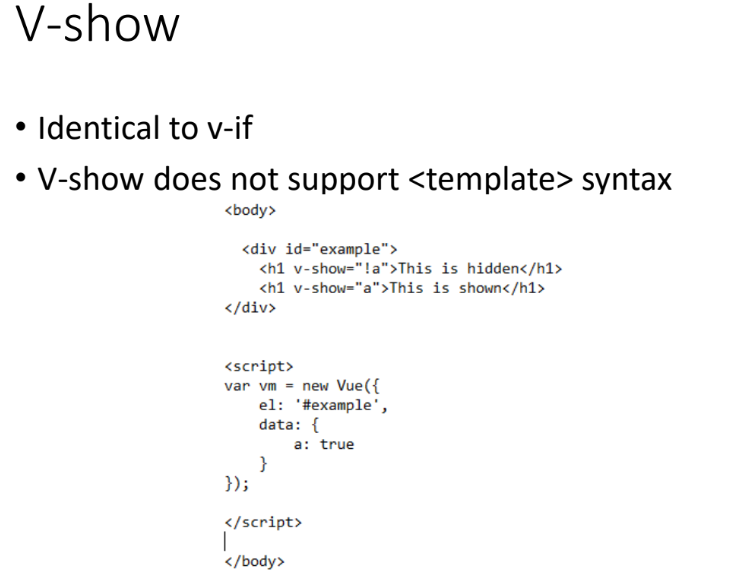


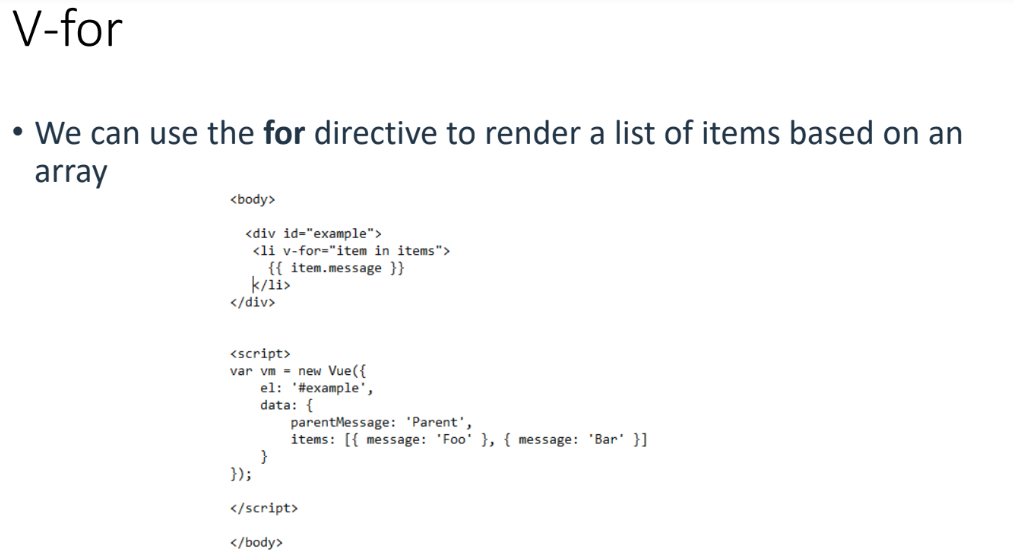
* 1. The **v-else** directive



* 1. The **v-else-if** directive



* 1. The **v-show** directive
  2. The **v-for** directive



**Binding html attributes**

• Html as many possible attributes.

Ex : an image tag have a src attribute, an a tag have a href attributes:



**Week 4 Summary**

Main topic

* User Input Handling with Vue

**Introduction**

• V- model directive is used to create two-way data bindings on form

input, textarea, and select elements

• It automatically picks the correct way to update the element based

on the input type.

• V-model will ignore the initial value, checked, or

selected attributes found on any form elements. It will always treat

the Vue instance data as the source of truth. You should declare the

initial value on the JavaScript side, inside the data option of your

component

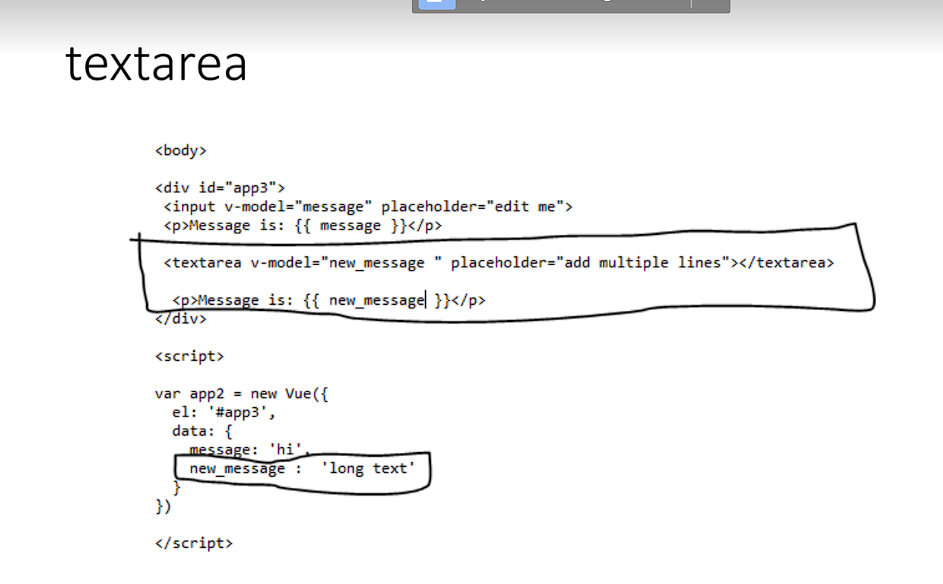
**V-model**

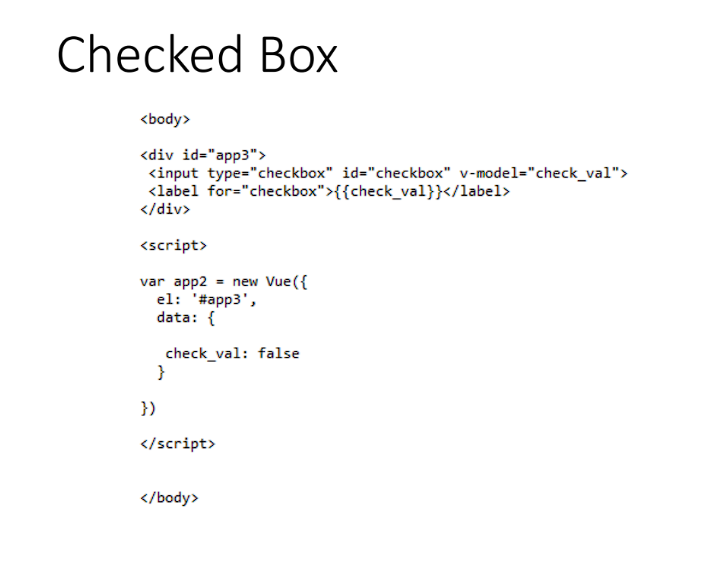
• V-model internally uses different properties and emits different

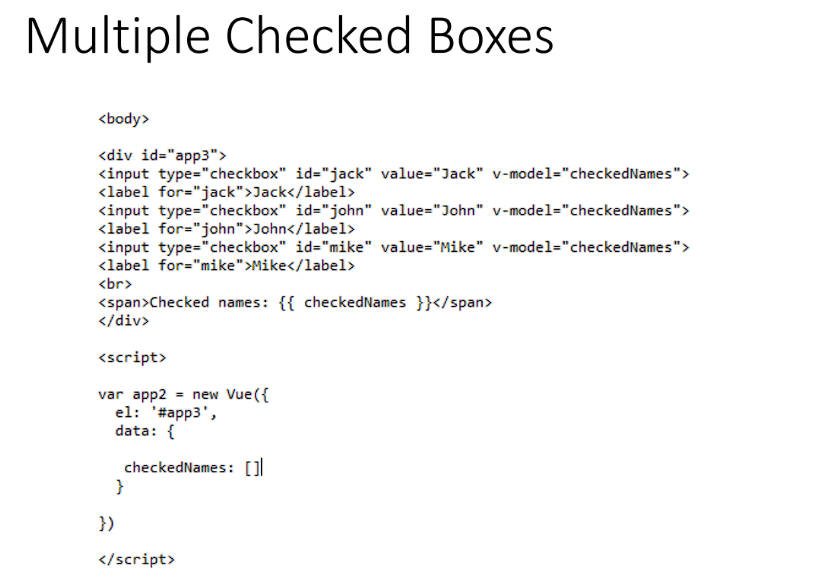
events for different input elements:

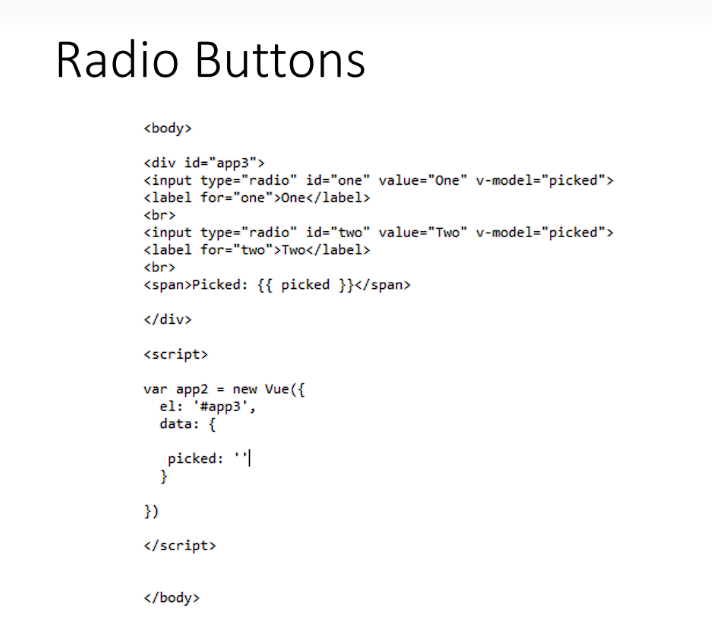
* Text and textarea elements use value property and input event;
* Checkboxes and radiobuttons use cheacked property and change event;
* Select fields use value as a prop and change as an event







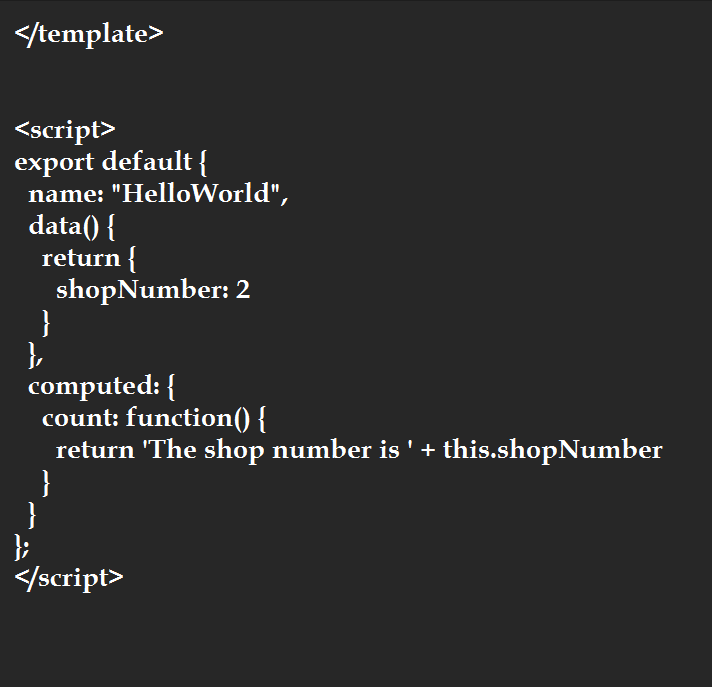




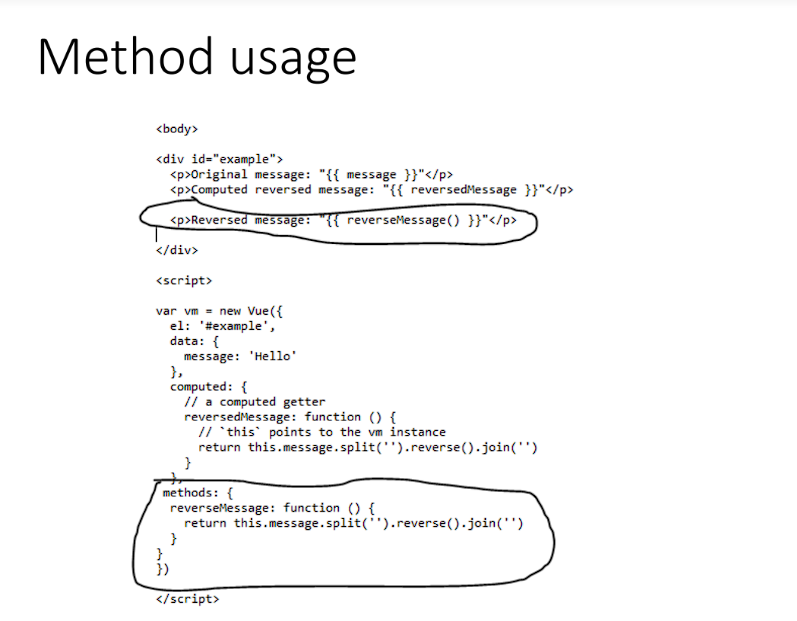
**Week 5 Summery**

In week 5 we learned about **computed properties ,** You can design a property using computed properties that can be used to effectively and readable modify, manipulate, and display data inside your components.

To calculate and display values based on a value or set of values in the data model, you can use computed properties. Additionally, it is possible for it to have some custom logic that is cached based on its dependencies; in other words, it doesn't need to reload but does have a dependency that changes, enabling it to detect changes and adjust its behavior in some way.



The count property receives the this.shopNumber as a dependency from the data function, then returns a sentence containing the this.shopNumber and it’s made available directly to the template.



**Week 6 summary**

Main Topic

Vue Components

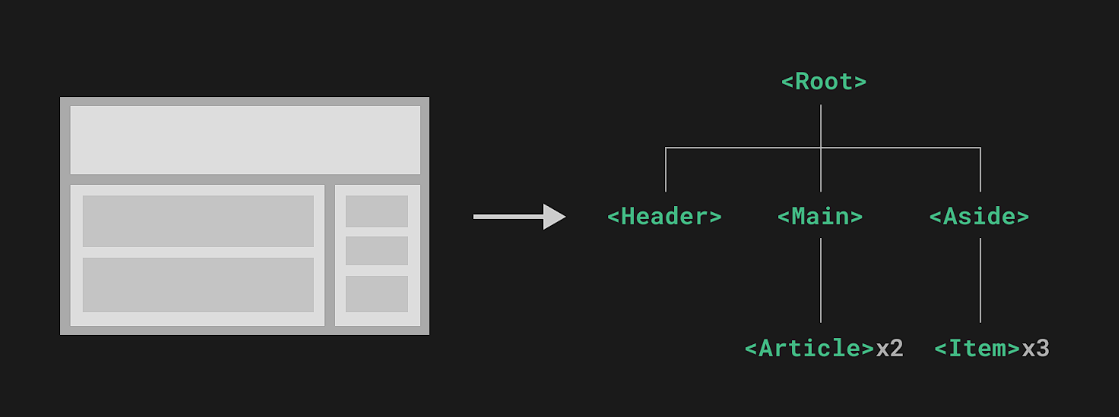
**Introduction**

• Components allow to split the UI into independent and reusable

pieces, and think about each piece in isolation.

• It's common for an app to be organized into a tree of nested

components:



**Dynamic Components**

• Dynamic components are created using the

keyword <component></component> and it is bound using a

property

**Reusing Components**

• Components can be reusable as many times as you want

• Each time you use a component, a new instance of it is created.

• When we defined the component, you may have noticed that data

wasn’t provided an object, like this:

• Instead, a component’s data option must be a function, so that

each instance can maintain an independent copy of the returned

data object



**Week 7 summery**

Main Topic

* Vue Components – Props

**Props**

• Props are used to pass down state to child components

• Props are the way components can accept data from components

that include them (parent components).

• When a component expects one or more prop, it must define them in

its props property

• Props are read-only and cannot be modified by the child component

because the parent component "owns" that value

export default {

props: ['foo'],

created() {

// props are exposed on `this`

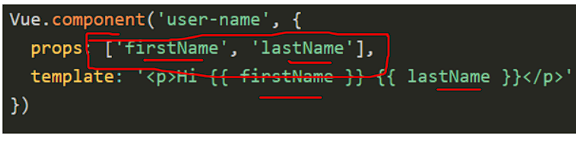
console.log(this.foo)

}

}

**Accepting multiple props**

• You can have multiple props by appending them to the array



**Setup prop type**

• You can specify the type of a prop by using an object instead of an

array, using the name of the property as the key of each property, and

the type as the value

• You can allow multiple different value types:

**Week 8 Summary**

Main topic

* Component Events

**Introduction**

• Many Vue patterns involve passing data from a parent component to its children using props

• But what if we need a child to affect its parent?

• Using emit, we can trigger events and pass data up the component hierarchy.

**Vue Emit**

• Vue $emit is a function that lets us emit, or send, custom events from a child component to its parent.

Handling Custom Event with Vue $Emit function – Example 01

• Intending to send a welcome message from our child component to the parent

|  |  |  |
| --- | --- | --- |
|  |  |  |
| The End.  Thank you |
|  |