**Slide 1:**

* Hi everyone, today our group would like to present to you a short analysis video about a software that we found very interest, Chromium. Our group member is Khai Phan, which is me speaking right now, Tyrell Unser and Mamadou Diallo.

**Slide 2:**

* In the video, I will briefly walk you through our analysis selection, our group project question: Which pattern are they using to write the code? Is there any requirement for others component to access it and why is the developer decided to use that?

**Slide 3:**

* The component that we are using for analyzing is keyed\_service and the reason why our group is choosing that is because one of the member while look at the component list and randomly click on that, and after a quick look, he told us that this could be a good starting point as it doesn’t have much files to read, there is documentation for every function on how to use it or what does this function do. Also, the design pattern using are clear, so we all agree and went from there.

**Slide 4:**

* One of the reason Chromium use keyed\_service is because of Apple iOS restriction on third-party browser installing from App Store. What that means is, browser cannot use the content layer as-is and must follow the component layer level between the iOS and the hardware. The browser can only request which layer of services/framework the application want to use and the OS will return exactly that service/framework, nothing more or less.
* One of the main reasons KeyedServices exist is to manage the service/profile that is currently running. Most of the time, the service/profile will depend on other services that is running for it to run. What KeyedServices is doing is provide a notification of LiveObject or DestroyedObject so that other services can be communicate, understand the status and situation, and know if it should run or not to avoid any unnecessary error.

**Slide 5:**

* This slide, we will discuss about the design pattern they are using.
* In the image, what keyed\_service component is offering there is multiple handling class, with the top class of BrowserContextDependencyManager to control the Web Context and the Keyed Services, which could be a framework, a service or an add-on that run to support the Web Context. Since there will be multiple keyed\_service and sometimes, they are depending on each other to run (like service1 must run in order service2 to run), a BrowserContextKeyedServiceFactory is required to keep track of all the work. A notifier class will be handling to keep track of the status for all service as in the event of 1 failure service could lead to the failure of other service (domino chain).
* Each of the keyed\_service can perform a check on another service/framework that it is depending on before running to ensure all requirement are satisfy. However, to do that, each keyed\_service must be a Singleton object. The reason for doing this is because each keyed\_service is unique, and it shouldn’t be redefined or re-initialize which change its state and value.
* Besides tracking if the services are running or not, the BrowserContextKeyedService can also perform a Shutdown and even a service Destroy on each of the Service in event of WebContext change.