Arquitectura de una aplicación en Angular

GeneXus™

Angular es un framework para construir aplicaciones web. Comencemos repasando algunos conceptos de estas aplicaciones.

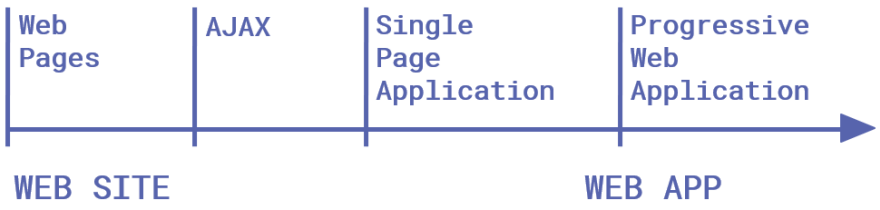
Arquitectura de una aplicación web

Las aplicaciones web tienen dos componentes, el front-end que ejecuta en el navegador web y el back-end que se ejecuta en un servidor web.

El front-end es donde corre realmente la aplicación y es la parte con la que el usuario interactúa, la que tiene la interface de usuario, se mantiene el estado de la aplicación y se maneja la lógica de negocios.

El back-end es donde están los servicios que sirven al front-end, se valida la lógica de negocios y se obtienen los datos de la base de datos.

Evolución de las aplicaciones web



Las aplicaciones web han tenido una evolución desde su primera implementación, de ser páginas casi estáticas a aplicaciones con una gran riqueza de contenido e interactividad.

Para aumentar la interactividad con los controles de la página, surgió la tecnología AJAX que permitía que algunas cosas se resolvieran en el cliente sin necesidad de que el pedido viaje al servidor.

Luego surgieron las aplicaciones de una sola página (SPA). Estas aplicaciones tienen la particularidad que, en respuesta a las interacciones del usuario, la página no se recarga nuevamente sino que solamente se actualiza la parte necesaria, lo que brinda una experiencia de usuario más fluida.

Y más recientemente aparecieron las aplicaciones web progresivas (PWA), que son aplicaciones web que permiten acceder a recursos de hardware y software de la máquina como si fuera una aplicación nativa y además permiten ser instaladas en el desktop o en dispositivos móviles.

¿Qué es Angular?

¿Y qué es Angular?

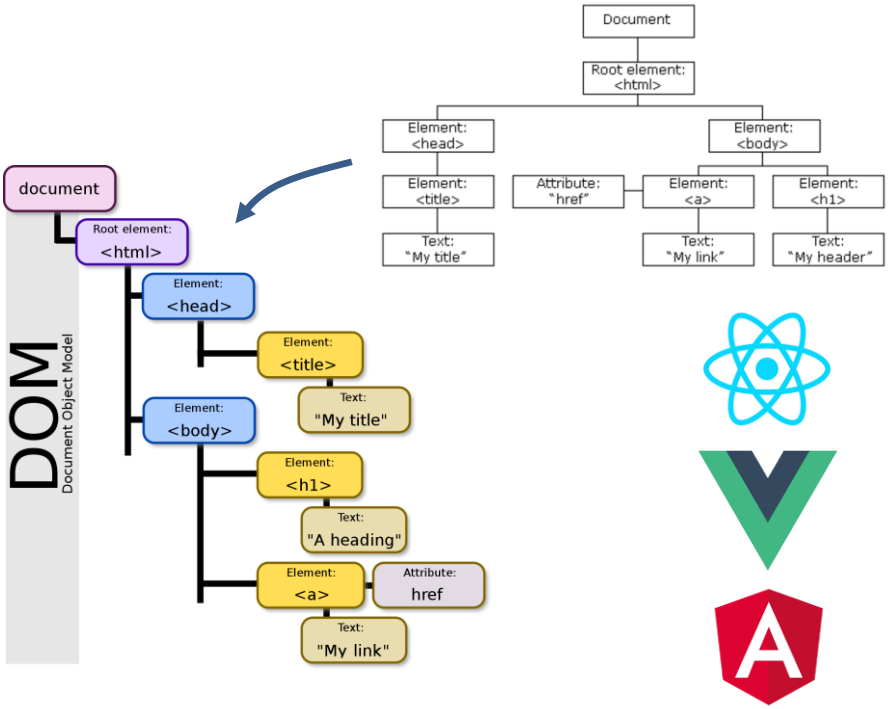
Angular es un framework de desarrollo para JavaScipt de código abierto creado por Google, que sirve para generar aplicaciones web de front-end, en particular aplicaciones de una sola página y aplicaciones web progresivas.

Angular se programa con TypeScript que es un superset de JavaScript que le agrega tipos de datos entre otras cosas, por lo que finalmente es JavaScript lo que se ejecuta en el navegador.

Este framework se especializa en mejorar la velocidad de dibujado de la página web en el browser, es decir la renderización, por lo que lo hace ideal para aplicaciones de front-end orientadas al usuario final, las aplicaciones denominadas “customer-facing”.

DOM: Document Object Model

```
1  <html>
2      <head>
3          <title>"My title"</title>
4      </head>
5      <body>
6          <h1>"A heading"</h1>
7          <a href="My link"></a>
8      </body>
9  </html>
```

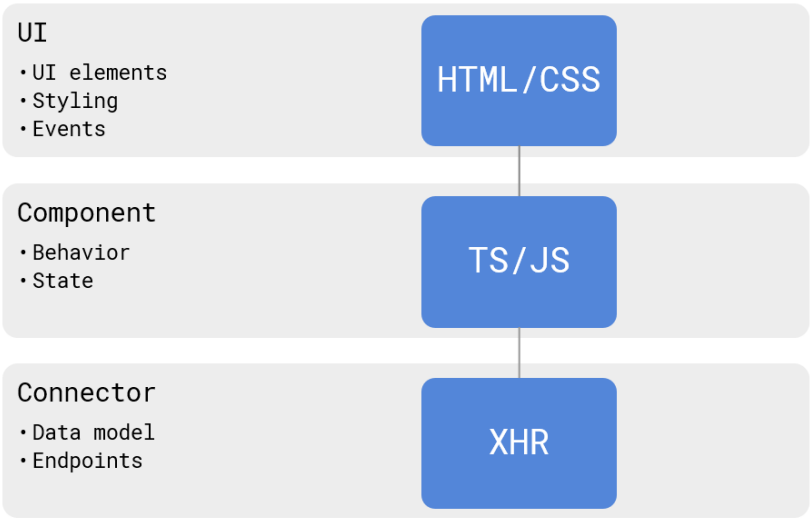


Para lograr que la pantalla se pinte rápidamente, Angular realiza un manejo eficiente del DOM (Document Object Model) que es la estructura jerárquica de objetos que genera el navegador cuando se carga un documento y se puede alterar mediante Javascript para cambiar dinámicamente los contenidos y aspecto de la página.

Como las aplicaciones de una sola página y las PWA, son pesadas y cada vez más complejas, el manejo eficiente del DOM es muy importante. Esto se logra mediante la virtualización del DOM y luego se actualiza el DOM real en forma muy rápida. Hay otros frameworks para front-end web como React o VueJS que resuelven también el manejo eficiente del DOM, pero solo abarcan una parte del desarrollo.

Angular en cambio es una solución completa, que además de la lógica para el manejo del DOM, incluye el manejo de componentes, librerías para navegar la aplicación, administración del estado de los controles en pantalla y mecanismos de comunicación entre el cliente y el servidor.

Estructura de una aplicación de front-end



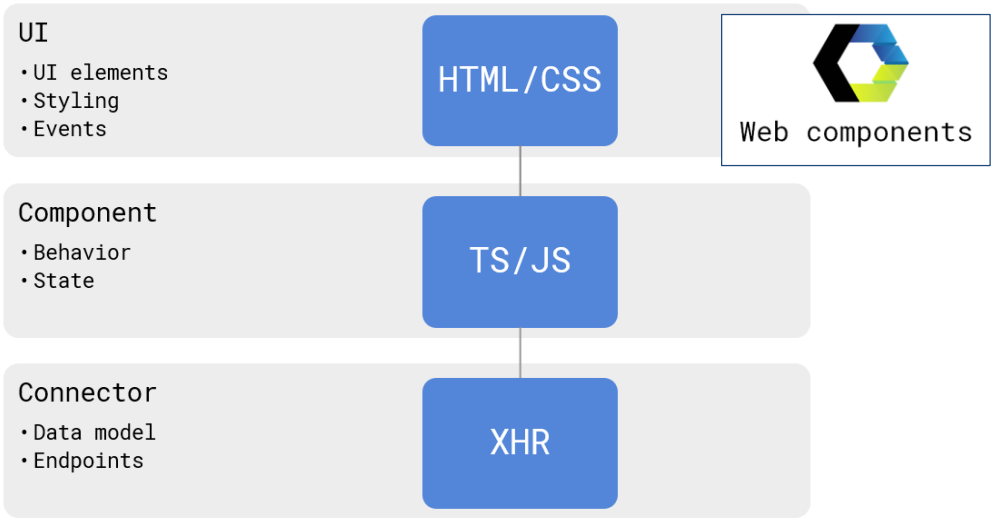
Una aplicación web de front-end tiene las siguientes 3 capas o elementos.

La User Interface, que es lo que el usuario ve. Son los elementos de pantalla o controles codificados en htmtl, con las definiciones de apariencia escritas en css (styling). Cuando el usuario interactúa con esta interfaz de usuario se generan eventos, que provocan cambios en el contenido de la página.

Component – Es el código que reacciona a los eventos de los elementos de la User Interface, es decir donde se programa el comportamiento de los controles de la UI. Esta capa también se encarga de mantener el estado de cada control y su código se programa en TypeScript o en JavaScript.

Connector – Es el representante del servidor en el cliente, lo que hace es pedirle al servidor datos y en algunos casos puede llegar a persistirlos, para minimizar las comunicaciones con el servidor.

Cómo se construye el front-end con el generador Angular de GeneXus



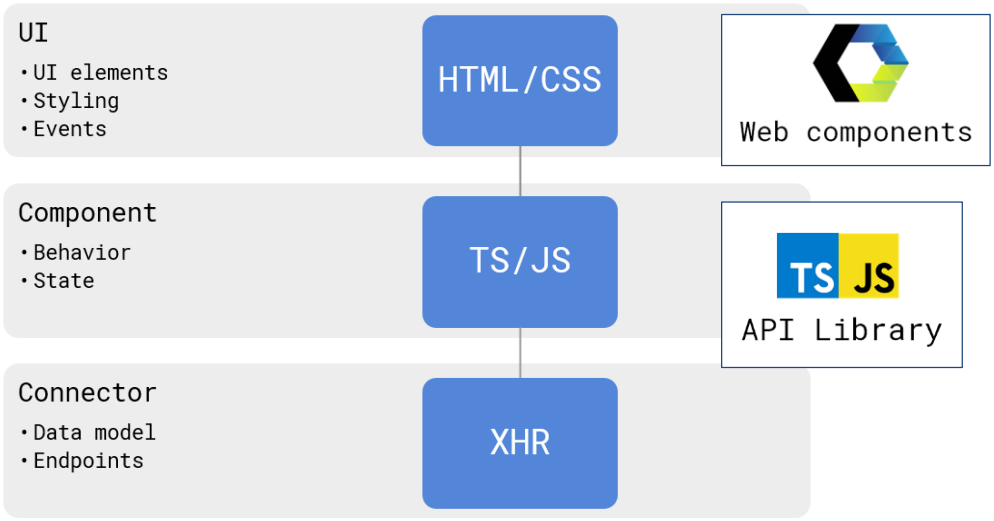
Veamos ahora cómo el generador Angular de GeneXus construye cada una de estas partes.

Para generar la User Interface se utilizan Web Components. Los web components son un estándar de W3C que nos permite encapsular html, javascript y css, en tags html definidos por nosotros.

Es decir que podemos extender el html y crear nuestros propios tags en base a elementos que corren nativamente en todos los browsers. Por ejemplo podemos crear un botón nuestro con su estilo y su comportamiento, que queda encapsulado en un web component y luego podemos usarlo en otros proyectos web o distribuirlo para que otros lo usen.

Es una nueva forma de componentizar en la web y hacer que los componentes sean portables.

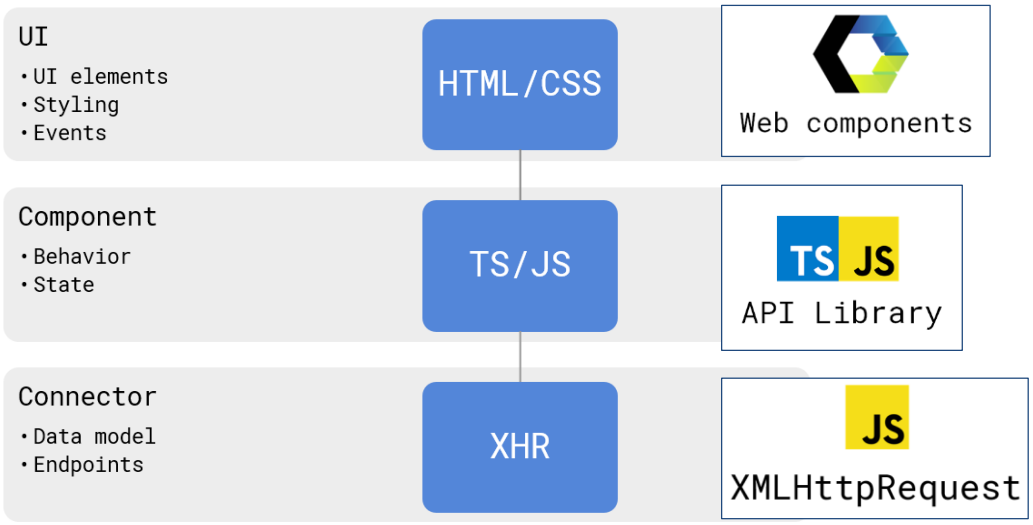
Cómo se construye el front-end con el generador Angular de GeneXus



Para implementar la capa de comportamiento se usa la GeneXus API Library.

Estas APIs son funciones de uso general como las de manejo de caracteres, funciones de fecha, etc., y que se generaban en el código específico de cada generador, ahora se implementaron en TypeScript y JavaScript para poder ser reutilizadas en todos los generadores, entre ellos Angular.

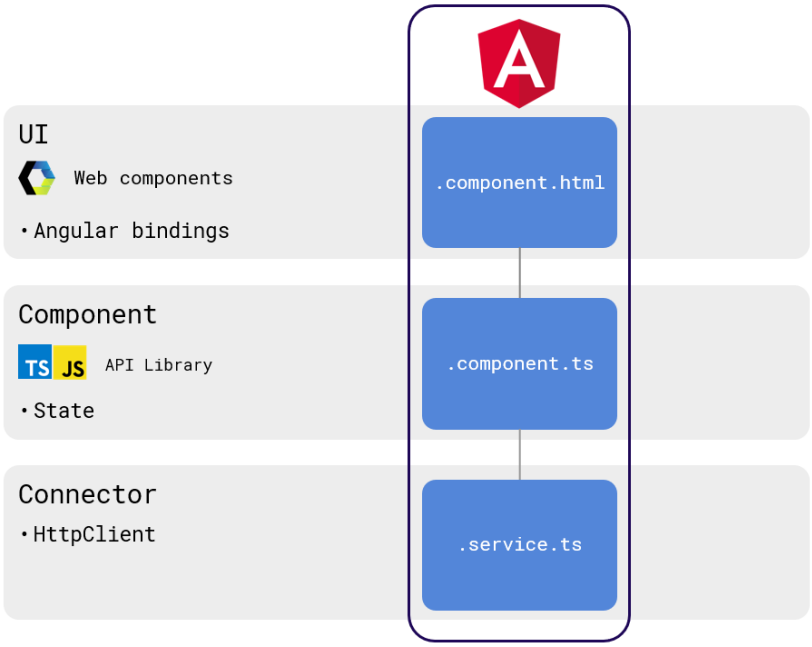
Cómo se construye el front-end con el generador Angular de GeneXus



Para el conector se utiliza el objeto XMLHttpRequest, que es un objeto JavaScript diseñado por Microsoft y adoptado por Mozilla, Apple y Google.
Actualmente es un estándar W3C y permite:

- Actualizar una página web sin recargar la página
- Solicitar datos al servidor y recibir datos del servidor, después de que se haya cargado la página
- Enviar datos al servidor, en segundo plano

Componente angular de la aplicación y archivos generados



Al bloque de construcción básico de una aplicación Angular se le denomina Componente. Cada componente consta de 3 partes que mapean las capas de una aplicación front-end y se representan en 3 archivos que son creados por el generador Angular.

Las definiciones de user interface se almacenan en un archivo cuyo nombre finaliza con .component.html. Este archivo es generado por GeneXus y contiene los web components que definen el template html y las definiciones del estilo de la User Interface. También se incluyen las asociaciones de los elementos de UI con los datos correspondientes de la base de datos.

El componente que define el comportamiento y estado de los controles se almacena en un archivo que finaliza con el nombre .component.ts. Allí están programados el comportamiento y estado de los controles en TypeScript, utilizando la GeneXus API Library.

El conector que se encarga de comunicarse con los servicios REST del servidor web se genera en un archivo que finaliza con el nombre .service.ts y GeneXus usa el HttpClient para implementar estas comunicaciones, utilizando el XMLHttpRequest.

Por cada objeto panel que se genere, se crearán estos 3 archivos cuyos nombres serán compuestos por el nombre del objeto que se ejecutará, seguido de las terminaciones que están en pantalla.

Escenarios de uso del generador Angular

- Desarrollo de una aplicación front-end web del tipo Customer-facing
- Cuando sabemos que vamos a necesitar una aplicación móvil nativa
- Convertir aplicaciones móviles nativas desarrolladas en GeneXus en una Web app
- Desarrollar back-end en GeneXus para front-end desarrollados con React, VueJs o Angular

Una pregunta que nos podemos hacer, es cuándo deberíamos elegir a Angular para generar nuestra aplicación web.

Un escenario posible es cuando necesitamos desarrollar una aplicación de front-end web orientado al cliente final (customer-facing). Como vimos Angular está pensado para tener una buena performance en estos casos de interfaces de usuario con requerimientos de mucha interactividad.

Otro caso en que el desarrollo en Angular es el indicado, es cuando estamos seguros que luego del front-end web, vamos a necesitar un front-end mobile nativo. En este caso los paneles que creamos para la aplicación Angular podrán ser reutilizados para generar la aplicación Android o iOS, definiéndose los layouts correspondientes a los dispositivos que vayamos a usar.

También es aplicable el uso de Angular cuando ya desarrollamos una aplicación nativa para Android o iOS con GeneXus y necesitamos un front-end web. Este caso es similar al anterior, ya que se pueden utilizar los mismos objetos, cambiando el diseño del layout para web y luego generando la misma aplicación que teníamos en Angular.

Un caso más en que es útil el uso de Angular como generador, es cuando queremos integrar aplicaciones codificadas manualmente en React, VueJs o Angular, con servicios o aplicaciones desarrolladas en GeneXus. Esto permite que si se cuenta con un equipo de desarrolladores especializados en el uso de estos frameworks, puedan seguir construyendo aplicaciones en su herramienta, pero toda la parte del back-end se puede desarrollar en GeneXus e integrarla al front-end.

En los siguientes videos veremos cómo podemos desarrollar aplicaciones web con el generador Angular en GeneXus.

training.genexus.com
wiki.genexus.com
training.genexus.com/certifications