Conceptos Clave de Angular para Entrevista

Este documento resume los conceptos más importantes de Angular que todo desarrollador debe conocer para una entrevista técnica. Incluye temas de arquitectura, ciclo de vida, seguridad, rendimiento y novedades recientes.

- **Arquitectura de Angular**: Basada en componentes, usa TypeScript, RxJS, HTML y CSS. Permite construir aplicaciones SPA altamente escalables. Soporta Stand-alone Components y NgModules.
- **Componentes y Templates**: Los componentes son la unidad básica de UI. Usan un decorador @Component, un template HTML y un archivo de estilos. Los templates pueden usar directivas y data binding.
- **Data Binding**: Incluye: Interpolación {{ }}, Property binding [ ], Event binding ( ), y Two-way binding [(ngModel)].
- **Directivas**: Instrucciones para manipular el DOM. Estructurales (*ngIf, *ngFor) y de atributo ([ngClass], [ngStyle]).
- **Servicios e Inyección de Dependencias**: Permiten lógica compartida entre componentes. Usan @Injectable y el sistema de inyección de Angular.
- **Routing y Lazy Loading**: Angular Router permite navegación entre vistas. Lazy loading carga módulos o componentes bajo demanda.
- **Ciclo de vida de un componente**: Métodos como ngOnInit, ngOnChanges, ngOnDestroy permiten reaccionar a eventos en el ciclo de vida.
- **RxJS y Programación Reactiva**: Angular usa Observables para manejar asincronía. Operadores como map, filter, switchMap son clave.
- **Formularios en Angular**: Dos enfoques: Template-driven y Reactive Forms. Validaciones síncronas y asíncronas.
- **Pipes**: Transforman datos en templates (ej: | date, | currency). Se pueden crear pipes personalizados.
- **Seguridad en Angular**: Protecciones contra XSS, sanitización automática, Content Security Policy, y Guards para rutas.
- **Optimización de rendimiento**: ChangeDetectionStrategy.OnPush, trackBy en *ngFor, lazy loading, preloading strategies, standalone components.
- **Testing en Angular**: Pruebas unitarias con Jasmine/Karma, pruebas end-to-end con Protractor o Cypress.
- **Novedades recientes**: Signals, nuevo control flow (@if, @for), mejoras en SSR, standalone APIs, build con esbuild y Vite.