API Report File for "@angular/common_upgrade"

> *Do not edit this file. It is a report generated by [API Extractor](#).*

```ts
import * as i0 from '@angular/core';
import * as i1 from '@angular/common';
import { InjectionToken } from '@angular/core';
import { Location as Location_2 } from '@angular/common';
import { LocationStrategy } from '@angular/common';
import { ModuleWithProviders } from '@angular/core';
import { PlatformLocation } from '@angular/common';
import { UpgradeModule } from '@angular/upgrade/static';

// @public
export class $locationShim {
    $$parse(url: string): void;
    $$parseLinkUrl(url: string, relHref?: string | null): boolean;
    constructor($injector: any, location: Location_2, platformLocation:
PlatformLocation, urlCodec: UrlCodec, locationStrategy: LocationStrategy);
    absUrl(): string;
    hash(): string;
    // (undocumented)
    hash(hash: string | number | null): this;
    host(): string;
    onChange(fn: (url: string, state: unknown, oldUrl: string, oldState: unknown) =>
void, err?: (e: Error) => void): void;
    path(): string;
    // (undocumented)
    path(path: string | number | null): this;
    port(): number | null;
    protocol(): string;
    replace(): this;
    search(): {
        [key: string]: unknown;
    };
    // (undocumented)
    search(search: string | number | {
        [key: string]: unknown;
    }): this;
    // (undocumented)
    search(search: string | number | {
        [key: string]: unknown;
    }, paramValue: null | undefined | string | number | boolean | string[]): this;
    state(): unknown;
    // (undocumented)
    state(state: unknown): this;
    url(): string;
    // (undocumented)
    url(url: string): this;
}
```

```
// @public
export class $locationShimProvider {
    $get(): $locationShim;
    constructor(ngUpgrade: UpgradeModule, location: Location_2, platformLocation:
PlatformLocation, urlCodec: UrlCodec, locationStrategy: LocationStrategy);
    hashPrefix(prefix?: string): void;
    html5Mode(mode?: any): void;
}

// @public
export class AngularJSUrlCodec implements UrlCodec {
    // (undocumented)
    areEqual(valA: string, valB: string): boolean;
    // (undocumented)
    decodeHash(hash: string): string;
    // (undocumented)
    decodePath(path: string, html5Mode?: boolean): string;
    // (undocumented)
    decodeSearch(search: string): {
        [k: string]: unknown;
    };
    // (undocumented)
    encodeHash(hash: string): string;
    // (undocumented)
    encodePath(path: string): string;
    // (undocumented)
    encodeSearch(search: string | {
        [k: string]: unknown;
    }): string;
    // (undocumented)
    normalize(href: string): string;
    // (undocumented)
    normalize(path: string, search: {
        [k: string]: unknown;
    }, hash: string, baseUrl?: string): string;
    // (undocumented)
    parse(url: string, base?: string): {
        href: string;
        protocol: string;
        host: string;
        search: string;
        hash: string;
        hostname: string;
        port: string;
        pathname: string;
    };
}

// @public
export const LOCATION_UPGRADE_CONFIGURATION: InjectionToken<LocationUpgradeConfig>;
```

```
// @public
export interface LocationUpgradeConfig {
    appBaseHref?: string;
    hashPrefix?: string;
    serverBaseHref?: string;
    urlCodec?: typeof UrlCodec;
    useHash?: boolean;
}

// @public
export class LocationUpgradeModule {
    // (undocumented)
    static config(config?: LocationUpgradeConfig):
ModuleWithProviders<LocationUpgradeModule>;
    // (undocumented)
    static ɵfac: i0.ɵɵFactoryDeclaration<LocationUpgradeModule, never>;
    // (undocumented)
    static ɵinj: i0.ɵɵInjectorDeclaration<LocationUpgradeModule>;
    // (undocumented)
    static ɵmod: i0.ɵɵNgModuleDeclaration<LocationUpgradeModule, never, [typeof
i1.CommonModule], never>;
}

// @public
export abstract class UrlCodec {
    abstract areEqual(valA: string, valB: string): boolean;
    abstract decodeHash(hash: string): string;
    abstract decodePath(path: string): string;
    abstract decodeSearch(search: string): {
        [k: string]: unknown;
    };
    abstract encodeHash(hash: string): string;
    abstract encodePath(path: string): string;
    abstract encodeSearch(search: string | {
        [k: string]: unknown;
    }): string;
    abstract normalize(href: string): string;
    abstract normalize(path: string, search: {
        [k: string]: unknown;
    }, hash: string, baseUrl?: string): string;
    abstract parse(url: string, base?: string): {
        href: string;
        protocol: string;
        host: string;
        search: string;
        hash: string;
        hostname: string;
        port: string;
        pathname: string;
    };
}
```

```
// (No @packageDocumentation comment for this package)
```