Address Class

Contains methods for accessing the component fields of address compound fields.

Namespace

System

Usage

Each of these methods is also equivalent to a read-only property. For each getter method, you can access the property using dot notation. For example, `myAddress.getCity()` is equivalent to `myAddress.city`.

You can't use dot notation to access compound fields' subfields directly on the parent field. Instead, assign the parent field to a variable of type `Address`, and then access its components. For example, to access the `City` field in `myAccount.BillingAddress`, do the following:

```
Address addr = myAccount.BillingAddress;
String acctCity = addr.City;
```

Important

"Address" in Salesforce can also refer to the Address standard object. When referencing the Address object in your Apex code, always use `Schema.Address` instead of `Address` to prevent confusion with the standard Address compound field. If referencing both the Address object and the Address standard field in the same snippet, you can differentiate between the two by using `System.Address` for the field and `Schema.Address` for the object.

Example

```
// Select and access Address fields.
// Call the getDistance() method in different ways.
Account[] records = [SELECT id, BillingAddress FROM Account LIMIT 10];
for(Account acct : records) {
   Address addr = acct.BillingAddress;
   Double lat = addr.latitude;
   Double lon = addr.longitude;
   Location loc1 = Location.newInstance(30.1944,-97.6682);
   Double apexDist1 = addr.getDistance(loc1, 'mi');
   Double apexDist2 = loc1.getDistance(addr, 'mi');
   System.assertEquals(apexDist1, apexDist2);
   Double apexDist3 = Location.getDistance(addr, loc1, 'mi');
   System.assertEquals(apexDist2, apexDist3);
}
```

- Address Methods

Returns the city field of this address.

- getCountry()
  Returns the text-only country/territory name component of this address.
- getCountryCode()
  Returns the country/territory code of this address if state and country/territory picklists are enabled in your organization. Otherwise, returns `null`.
- getDistance(toLocation, unit)
  Returns the distance from this location to the specified location using the specified unit.
- getGeocodeAccuracy()
  When using geolocation data for a given address, this method gives you relative location information based on latitude and longitude values. For example, you can find out if the latitude and longitude values point to the middle of the street, instead of the exact address.
- getLatitude()
  Returns the latitude field of this address.
- getLongitude()
  Returns the longitude field of this address.
- getPostalCode()
  Returns the postal code of this address.
- getState()
  Returns the text-only state name component of this address.
- getStateCode()
  Returns the state code of this address if state and country/territory picklists are enabled in your organization. Otherwise, returns `null`.
- getStreet()
  Returns the street field of this address.

getCity()

Returns the city field of this address.

Signature

```
public String getCity()
```

Return Value

Type: String

getCountry()

Returns the text-only country/territory name component of this address.

Signature

```
public String getCountry()
```

Return Value

Type: String

getCountryCode()

Returns the country/territory code of this address if state and country/territory picklists are enabled in your organization. Otherwise, returns `null`.

Signature

getDistance(toLocation, unit)

Returns the distance from this location to the specified location using the specified unit.

Signature

```
public Double getDistance(Location toLocation, String unit)
```

Parameters

toLocation

Type: Location

The `Location` to which you want to calculate the distance from the current `Location`.

unit

Type: String

The distance unit you want to use: `mi` or `km`.

Return Value

Type: Double

getGeocodeAccuracy()

When using geolocation data for a given address, this method gives you relative location information based on latitude and longitude values. For example, you can find out if the latitude and longitude values point to the middle of the street, instead of the exact address.

Signature

```
public String getGeocodeAccuracy()
```

Return Value

Type: String

The `getGeocodeAccuracy()` return value tells you more about the location at a latitude and longitude for a given address. For example, `Zip` means the latitude and longitude point to the center of the zip code area, in case a match for an exact street address can't be found.

Accuracy Value	Description
`Address`	In the same building
`NearAddress`	Near the address
`Block`	Midway point of the block
`Street`	Midway point of the street
`ExtendedZip`	Center of the extended zip code area
`Zip`	Center of the zip code area
`Neighborhood`	Center of the neighborhood

`County`	Center of the county
`State`	Center of the state
`Unknown`	No match for the address was found

Geocodes are added only for some standard addresses.

- `Billing Address` on accounts
- `Shipping Address` on accounts
- `Mailing Address` on contacts
- `Address` on leads

Person accounts are not supported.

> **Note**
>
> For `getGeocodeAccuracy()` to work, set up and activate the geocode data integration rules for the related address fields.

getLatitude()

Returns the latitude field of this address.

Signature

```
public Double getLatitude()
```

Return Value

Type: Double

getLongitude()

Returns the longitude field of this address.

Signature

```
public Double getLongitude()
```

Return Value

Type: Double

getPostalCode()

Returns the postal code of this address.

Signature

```
public String getPostalCode()
```

Return Value

Type: String

getState()

Return Value

Type: String

getStateCode()

Returns the state code of this address if state and country/territory picklists are enabled in your organization. Otherwise, returns `null`.

Signature

```
public String getStateCode()
```

Return Value

Type: String

getStreet()

Returns the street field of this address.

Signature

```
public String getStreet()
```

Return Value

Type: String