Custom Address Fields Developer Guide

Version 68.0, Winter ’27

Last updated: October 9, 2026

© Copyright 2000–2026 Salesforce, Inc. All rights reserved. Salesforce is a registered trademark of Salesforce, Inc., as are other names and marks. Other marks appearing herein may be trademarks of their respective owners.

CONTENTS

Chapter 1: Custom Address Fields 1

Chapter 2: Custom Address Fields Requirements and Limitations 2

Chapter 3: Configure State and Country/Territory Picklists 5

Chapter 4: Enable Custom Address Fields 6

Chapter 5: Add a Geocode to a Custom Address Field 7

Chapter 6: Apex Examples 10

Chapter 7: Metadata API Example 13

Chapter 8: REST API Examples 14

Chapter 9: SOAP API Examples 17

Chapter 10: Tooling API Examples 21

CHAPTER 1 Custom Address Fields

Use the Address field type to create custom fields that store address data in a structured compound data type. Compound fields are an abstraction that can simplify application code that handles the values, leading to more concise, understandable code. With Custom Address Fields, custom addresses are accessible as a single, structured field, or as individual component fields.

EDITIONS

Available in: all editions

Note: Before you enable custom address fields, review the Custom Address Fields Requirements and Limitations. To discuss the feature and ask questions, join the Custom Address Fields Discussion group on the Trailblazer Community.

Address compound fields are delivered for standard fields on standard objects. Now with Custom Address Fields, custom fields can mirror the standard address field behavior. End users can add and retrieve address data via these custom Address fields on standard and custom objects. Users can edit the custom address field data in records and view custom address data in list views and reports.

Note: For custom compound fields, each component counts as one custom field toward your org's allocations. Thus each custom address field counts as nine custom fields: one each for street, city, postal code, country code, state code, geocode accuracy level, longitude, and latitude, plus one for internal use. For more information on the allocations for your org, see Salesforce Features and Edition Allocations in Salesforce Help.

CHAPTER 2 Custom Address Fields Requirements and Limitations

Before you enable Custom Address Fields, configure State and Country/Territory picklists and review the limitations of this feature.

Note: To discuss the feature and ask questions, join the Custom Address Fields Discussion group on the Trailblazer Community.

EDITIONS

Available in: all editions.

Custom Address Fields Requirement: State and Country/Territory Picklists

Custom Address Fields uses picklists for the State and Country address fields. For more information, see Configure State and Country/Territory Picklists.

Custom Address Fields Requirement: Package Deployment

If a package contains a custom field with the Address field type, package deployment requires that Custom Address Fields is enabled in the target org.

Custom Address Fields and Org Limits

For custom compound fields, each component counts as one custom field toward your org's allocations. Thus each custom address field counts as nine custom fields: one each for street, city, postal code, country code, state code, geocode accuracy level, longitude, and latitude, plus one for internal use. For more information on the allocations for your org, see Salesforce Features and Edition Allocations in Salesforce Help.

Limitations for Custom Address Fields

Before you enable Custom Address Fields or add a custom address field, understand the limitations of this feature.

These items aren't supported with custom address fields.

- The conversion of address data into custom fields of type Address from custom fields of other types.
- Approvals
- Data Import Wizard
- Fuzzy matching
- Composite API

- Field Encryption
- Field Sets
- Flow Screen Input Component: Address
- Lead Conversion
- Lightning Web Components
- Merge Fields
- Search, including global search, lookup search, SOSL queries, and Search Manager
- Visualforce pages
- Workflow

Salesforce hasn't validated custom address fields with these items.

- Schema Builder
- Web-to-Case and Email-to-Case
- Generating Leads from Your Website
- Filtering in a related list
- Bulk API 1.0
- Data Loader

This functionality is either unavailable or limited with Custom Address Fields.

- As with standard address fields, you can't mark a custom address field as required.
- Data Export doesn't support records with a custom address field whose `GeocodeAccuracy` field contains a value. The export succeeds when the `GeocodeAccuracy` field is empty.
- You can't use the DISTANCE function with a custom address field.
- To export data stored in custom fields of type Address, use API or SOQL queries. Bulk API doesn't support the export of custom compound fields.
- The error message when you attempt to export a custom address field with Bulk API incorrectly states that the functionality isn't enabled. Bulk API doesn't support the export of custom compound fields.
- To populate a custom address field with imported data, use REST API or Bulk API 2.0.
- Search queries only support the data stored within the Street component of custom fields of type Address. The State, City, Postal Code, and Country components aren't supported for search.
- In Skinny Tables, you can't select a component of a custom address field as partition column.
- When configuring search results for an object, custom address fields aren't supported in Search Filter Fields (only available in Salesforce Classic). If you specify a custom address field as a Search Filter Field in a search layout, package installation and Metadata `deploy()` fails.
- Compound address fields aren't supported in reports. To include a custom address field in a report, add the individual address components, such as street, city, state, and zip.
- When using a custom address field in a Data Integration Rule, the Country and State components are unavailable for field mapping.
- You can't rename the labels for the individual components of a custom address field.
- You can localize the label of a custom address field. However, you can't localize the labels of the individual components within a custom address field.
- The word "Address" isn't appended to the section label for a custom address field. If you include the word "Address" in the field label, it's included in the label for every component. For example,

"Warehouse Address (State)" instead of "Warehouse (State)". These labels are inconsistent with the label behavior for standard address fields.

- The length of the GeoCodeAccuracy field for custom fields of data type Address isn't consistent with standard field of type Address.

CHAPTER 3 Configure State and Country/Territory Picklists

Custom Address Fields uses picklists for the State and Country address fields. Before you enable custom address fields, configure State and Country/Territory picklists.

EDITIONS

Available in: all editions except Database.com.

Note: Before you enable custom address fields, review the Custom Address Fields Requirements and Limitations. To discuss the feature and ask questions, join the Custom Address Fields Discussion group on the Trailblazer Community.

If State and Country/Territory picklists are enabled, those picklist values are used in standard address fields. With Custom Address Fields, the same picklist values are automatically available in custom address fields. You can't specify separate picklist values for standard and custom address fields.

If State and Country/Territory Picklists aren't enabled, those picklists are enabled for custom address fields with Custom Address Fields. By default, all countries, territories, and their states and provinces are visible to users. To specify the available picklist values in Salesforce, configure State and Country/Territory Picklists.

When you configure these picklist values, the behavior of standard address fields is unaffected unless you enable State and Country/Territory Picklists for standard fields through Setup. Enabling the picklists for standard fields isn't required to use Custom Address Fields.

To configure the picklists, use the AddressSettings Metadata API or see Configure State and Country/Territory Picklists in Salesforce Help.

For details on enabling the picklists for standard address fields, see Let Users Select States, Countries, and Territories from Picklists in Salesforce Help.

CHAPTER 4 Enable Custom Address Fields

After you review the feature limitations and configure the State and Country/Territory picklists, enable the Custom Address Fields feature.

Note: To discuss the feature and ask questions, join the Custom Address Fields Discussion group on the Trailblazer Community.

Before you enable custom address fields, review the Custom Address Fields Requirements and Limitations and Configure State and Country/Territory Picklists.

To enable Custom Address Fields in Setup:

1. In Setup, in the Quick Find box, enter *User Interface*, and then select **User Interface**.
2. In the Setup section, select **Use custom address fields** and save your changes.

   After you enable custom address fields, the Address data type is available when you add a field via Object Manager.

   Note: This feature can't be disabled.

To enable Custom Address Fields via Metadata API, use the `enableCustomAddressField` field in the CustomAddressFieldSettings metadata type.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: all editions

USER PERMISSIONS

To modify user interface settings:

- Customize Application

CHAPTER 5 Add a Geocode to a Custom Address Field

The method to get geocodes differs between standard and custom address fields. To give your users precise geographical information, add geocode information to a custom address field with Apex, Visualforce, and a map API.

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience
Available in: all editions

User Permissions Needed	
To modify user interface settings:	Customize Application

1. Create an Apex class that retrieves latitude and longitude from your preferred map API. This example calls the Google Map API as defined in the String variable `endpoint`.

```
public class GeoCodeExample {
    @future(callout=true)
    public static void parseJSONResponse() {
        double lat;
        double lng;
        String city = null;
        String street = null;
           String stateCode = null;
        String countryCode = null;

        Account record = [SELECT Mailing_Address__c FROM Account
WHERE Id = 'Account ID'];
        Address customAddress = record.Mailing_Address__c;

        //Remove white spaces from address components
        if(customAddress.getCity() != null){
             city = customAddress.getCity().deleteWhitespace();
        }
        if(customAddress.getStreet() != null){
           street = customAddress.getStreet().deleteWhitespace();

        }
        if(customAddress.getStateCode() != null){
             stateCode = customAddress.getStateCode();
        }
        if(customAddress.getCountryCode() != null){
             countryCode = customAddress.getCountryCode();
        }

        //concatenate strings
        String address = street+city+stateCode+countryCode;
```

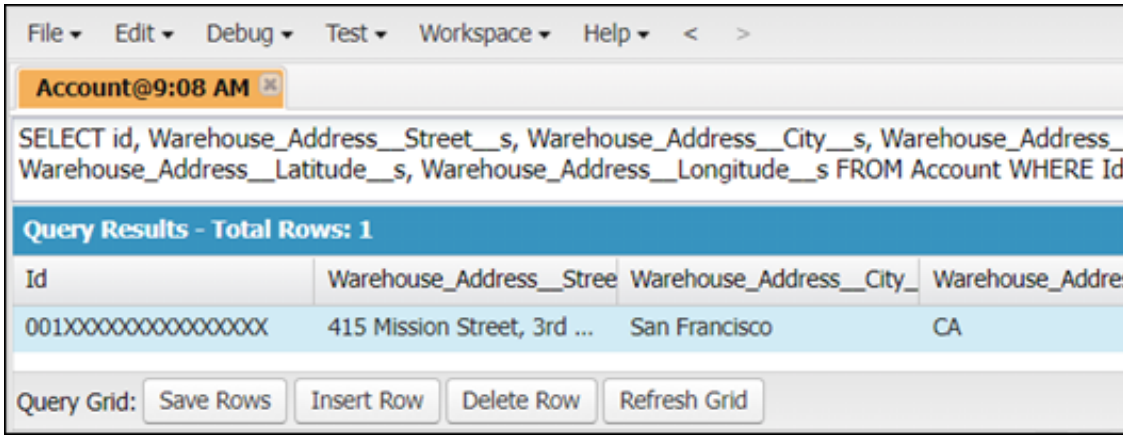
```
        String key='API key';
        Http httpProtocol = new Http();
        // Create HTTP request to send.
        HttpRequest request = new HttpRequest();
        // Set the endpoint URL.
        // USING GOOGLE MAP API
        String endpoint =
'https://maps.googleapis.com/maps/api/geocode/json?address='+address+'&key='+key;

        request.setEndPoint(endpoint);
        // Set the HTTP verb to GET.
        request.setMethod('GET');
        // Send the HTTP request and get the response.
        // The response is in JSON format.
        HttpResponse response = httpProtocol.send(request);

       // Parse JSON response to get all the totalPrice field
values.
       JSONParser parser = JSON.createParser(response.getBody());


        while (parser.nextToken() != null) {
           if ((parser.getCurrentToken() == JSONToken.FIELD_NAME)
 &&
                (parser.getText() == 'lat')) {
                parser.nextToken();
                // Get latitude
                lat = parser.getDoubleValue();

                parser.nextToken();
                parser.nextToken();
                //Get longitude
                lng = parser.getDoubleValue();
                 }
              }
        // Update lat long of account record
        record.Mailing_Address__Latitude__s=lat;
        record.Mailing_Address__Longitude__s=lng;
        update record;
    }
}
```

2. Create a Visualforce page that triggers the geocode service from the map API.

```
<apex:page id="pg" controller="GeoCodeExample">
<apex:form >
    <apex:pageBlock id="pb">
        <apex:pageBlockButtons >
            <apex:commandButton value="Get GeoCode For Custom
Address Field"
            action="{!parseJSONResponse}"/>
        </apex:pageBlockButtons>
    </apex:pageBlock>
```

```
</apex:form>
</apex:page>
```

3. On the Visualforce page, click **Get GeoCode For Custom Address Field** to trigger the code. To see the latitude and longitude values populated, query the account information in Developer Console.



To automate the process of updating custom address fields with latitude and longitude, set up a trigger to invoke the Apex class.

Note: The example in this topic uses a third-party map API to retrieve latitude and longitude. Using a Salesforce trigger to invoke this Apex class calls the map API each time the class is invoked. This action can result in charges from your API provider.

CHAPTER 6 Apex Examples

Apex code examples for Custom Address Fields. The examples create a record with custom address data, update the custom address on an existing record, and delete a record that contains custom address data.

Note: Before you create a custom address field, review the Custom Address Fields Requirements and Limitations. To discuss the feature and ask questions, join the Custom Address Fields Discussion group on the Trailblazer Community.

EDITIONS

Available in: all editions.

Insert a Record

This example code creates an Opportunity record which includes address data stored in the custom address field, "Mailing Address".

```
Opportunity a = new Opportunity();
a.StageName='Prospecting';
a.CloseDate=System.today();
a.Name = 'Dublin Order';
a.Mailing_Address__StateCode__s='CA';
a.Mailing_Address__CountryCode__s='US';
a.Mailing_Address__Street__s='1234 Dublin Blvd';
a.Mailing_Address__PostalCode__s='12345';
a.Mailing_Address__City__s='Dublin';
a.Mailing_Address__Latitude__s=80.34;
a.Mailing_Address__Longitude__s=80.35;
a.Mailing_Address__GeocodeAccuracy__s='Address';
insert a;
```

This example code add a record for a custom object, "Gas Station" (Gas_Station__c). The new record includes address data stored in the the custom address field, "Mailing Address".

```
Gas_Station__c a = new Gas_Station__c();
a.Name = 'Amador Valley';
a.Mailing_Address__StateCode__s='CA';
a.Mailing_Address__CountryCode__s='US';
a.Mailing_Address__Street__s='1234 Dublin Blvd';
a.Mailing_Address__PostalCode__s='12345';
a.Mailing_Address__City__s='Dublin';
a.Mailing_Address__Latitude__s=80.34;
a.Mailing_Address__Longitude__s=80.35;
a.Mailing_Address__GeocodeAccuracy__s='Address';
insert a;
```

Update an Existing Record

This example code updates the custom address field "Mailing Address" on an Opportunity record with ID 006XXXXXXXXXXXXXX.

```
Opportunity o = [select Id from Opportunity where
Id='006XXXXXXXXXXXXXX'];
o.Mailing_Address__StateCode__s='CA';
o.Mailing_Address__CountryCode__s='US';
o.Mailing_Address__Street__s='1234 Dublin Blvd';
o.Mailing_Address__PostalCode__s='12345';
o.Mailing_Address__City__s='Dublin';
o.Mailing_Address__Latitude__s=80.34;
o.Mailing_Address__Longitude__s=80.35;
o.Mailing_Address__GeocodeAccuracy__s='Address';
update o;
```

This example code updates an existing record for a custom object, "Gas Station" (Gas_Station__c) with ID aIsXXXXXXXXXXXXXXX. It updates custom address field "Mailing Address".

```
Gas_Station__c a = [select Id from Gas_Station__c where
Id='aIsXXXXXXXXXXXXXXX'];
a.Mailing_Address__StateCode__s='CA';
a.Mailing_Address__CountryCode__s='US';
a.Mailing_Address__Street__s='1234 Dublin Blvd';
a.Mailing_Address__PostalCode__s='12345';
a.Mailing_Address__City__s='Dublin';
a.Mailing_Address__Latitude__s=80.34;
a.Mailing_Address__Longitude__s=80.35;
a.Mailing_Address__GeocodeAccuracy__s='Address';
update a;
```

Delete Data Within a Custom Address Field from a Record

To delete an address stored in a custom address field from a record, update the record. This example code removes the data stored the custom address field "Mailing Address" on an Opportunity record with ID 006XXXXXXXXXXXXXX.

```
Opportunity o = [select Id from Opportunity where
Id='006XXXXXXXXXXXXXX'];
o.Mailing_Address__StateCode__s= null;
o.Mailing_Address__CountryCode__s= null;
o.Mailing_Address__Street__s=null;
o.Mailing_Address__PostalCode__s=null;
o.Mailing_Address__City__s=null;
o.Mailing_Address__Latitude__s=null;
o.Mailing_Address__Longitude__s=null;
o.Mailing_Address__GeocodeAccuracy__s=null;
update o;
```

Delete a Record

This code deletes a record for the custom object, "Gas Station" (Gas_Station__c) with ID aIsXXXXXXXXXXXXXX. When a record is deleted, all data for that record is deleted, including the custom address field information.

```
Gas_Station__c a = [select Id from Gas_Station__c where
Id='aIsXXXXXXXXXXXXXX'];
delete a;
```

CHAPTER 7 Metadata API Example

To create a custom address field on an object, use Metadata API.

Note: Before you create a custom address field, review the Custom Address Fields Requirements and Limitations. To discuss the feature and ask questions, join the Custom Address Fields Discussion group on the Trailblazer Community.

This example creates a custom object of type Address on the Account object.

```
<?xml version="1.0" encoding="UTF-8"?>
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">

    <fields>
        <fullName>MailingAddress__c</fullName>
        <externalId>false</externalId>
        <label>Mailing Address</label>
        <required>false</required>
        <type>Address</type>
        <unique>false</unique>
   </fields>
 </CustomObject>
```

CHAPTER 8 REST API Examples

Use REST API to create, update, or delete a record with Custom Address Fields data.

Note: Before you create a custom address field, review the Custom Address Fields Requirements and Limitations. To discuss the feature and ask questions, join the Custom Address Fields Discussion group on the Trailblazer Community.

Create a New Account with Data in a Custom Address Field

To create a new record, use the sObject resource. You supply the required field values in the request data, and send the request using the POST HTTP method. The response body contains the ID of the new record if the call is successful.

This example creates an Account record which includes address data stored in the Mailing Address custom address field.

Example HTTP POST method to create a new Account

```
curl
https://MyDomainName.my.salesforce.com/services/data/68.0/sobjects/Account
 -H "Authorization: Bearer token" -H "Content-Type: application/json"
 -d "@newaccount.json"
```

Example request body newaccount.json file to create a new Account

```
{
"Name" : "Acme Incorporated",
"Mailing_Address__City__s" : "Ahmedabad",
"Mailing_Address__CountryCode__s" : "IN",
"Mailing_Address__Street__s" : "102 Suryakoti",
"Mailing_Address__PostalCode__s" : "380022",
"Mailing_Address__StateCode__s": "GJ",
"Mailing_Address__Latitude__s" : "37.775",
"Mailing_Address__Longitude__s" : "-122.418",
"Mailing_Address__GeocodeAccuracy__s" : "Address"
}
```

Example response body after successfully creating a new Account

```
{
  "id" : "001XXXXXXXXXXXXXX",
  "errors" : [ ],
  "success" : true
}
```

Update Data Within a Custom Address Field on a Record

To update a record, use the sObject Rows resource. Provide the updated record information in your request data and use the PATCH method of the resource with a specific record ID to update that record. Records in a single file must be of the same object type.

This example updates the data stored in the Mailing Address custom address field for record ID `001XXXXXXXXXXXXXX`.

HTTP PATCH example for updating an Account

```
curl
https://MyDomainName.my.salesforce.com/services/data/68.0/sobjects/Account/001XXXXXXXXXXXXX
 -H "Authorization: Bearer token" -H "Content-Type: application/json"
 -d @patchaccount.json -X PATCH
```

Example request body patchaccount.json file for updating the custom field, Mailing Address, on an Account

```
{
"Mailing_Address__City__s" : "Surendranagar",
"Mailing_Address__CountryCode__s" : "IN",
"Mailing_Address__Street__s" : "20 Udhyog Nagar",
"Mailing_Address__PostalCode__s" : "363001",
"Mailing_Address__StateCode__s":  "GJ",
"Mailing_Address__Latitude__s" : "22.757580",
"Mailing_Address__Longitude__s" : "71.619350",
"Mailing_Address__GeocodeAccuracy__s" : "Address"
}
```

Example response body after successfully updating an Account: None returned

Delete Data Within a Custom Address Field on a Record

To delete address data stored within a custom address field on a record, update the record with the sObject Rows resource. Provide the updated record information in your request data and use the PATCH method of the resource with a specific record ID to update that record. Records in a single file must be of the same object type.

This example deletes the data stored in the Mailing Address custom address field for record ID `001XXXXXXXXXXXXXX`.

HTTP PATCH example for updating an Account

```
curl
https://MyDomainName.my.salesforce.com/services/data/68.0/sobjects/Account/001XXXXXXXXXXXXX
 -H "Authorization: Bearer token" -H "Content-Type: application/json"
 -d @patchaccount.json -X PATCH
```

Example request body patchaccount.json file for updating the custom field, Mailing Address, on an Account

```
{
"Mailing_Address__City__s" : null,
"Mailing_Address__CountryCode__s" : null,
"Mailing_Address__Street__s" : null,
"Mailing_Address__PostalCode__s" : null,
"Mailing_Address__StateCode__s":  null,
"Mailing_Address__Latitude__s" : null,
"Mailing_Address__Longitude__s" : null,
"Mailing_Address__GeocodeAccuracy__s" : null
}
```

Example response body after successfully updating an Account: None returned

Delete a Record That Contains Data in a Custom Address Field

To delete records, use the sObject Rows resource. Specify the record ID and use the DELETE method of the resource to delete a record. When a record is deleted, all data for that record is deleted, including the custom address field information.

This example deletes the Account record with ID `001XXXXXXXXXXXXXXX`.

HTTP DELETE example for updating an Account

```
curl
https://MyDomainName.my.salesforce.com/services/data/68.0/sobjects/Account/001XXXXXXXXXXXXXX
  -H "Authorization: Bearer token" -X DELETE
```

Example request body: None needed

Example response body after successfully updating an Account: `200 OK`

CHAPTER 9 SOAP API Examples

Use SOAP API to create, update, or delete a record with Custom Address Fields data.

Note: Before you create a custom address field, review the Custom Address Fields Requirements and Limitations. To discuss the feature and ask questions, join the Custom Address Fields Discussion group on the Trailblazer Community.

Create a New Account with Data in a Custom Address Field

This example creates an Account record which includes address data stored in the Mailing Address custom address field.

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:urn="urn:enterprise.soap.sforce.com"
    xmlns:urn1="urn:sobject.enterprise.soap.sforce.com"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Header>
        <urn:SessionHeader>
            <urn:sessionId>$0XXXXXXXXXXXXXXX</urn:sessionId>
        </urn:SessionHeader>
    </soapenv:Header>
    <soapenv:Body>
        <urn:create>
            <urn:sObjects xsi:type="urn1:Account"> <!--Zero or more
repetitions:-->
                <Name>Puneet Ahmedabad Account</Name>

<Mailing_Address__City__s>Ahmedabad</Mailing_Address__City__s>
                <Mailing_Address__Street__s>102
Suryakoti</Mailing_Address__Street__s>

<Mailing_Address__PostalCode__s>380022</Mailing_Address__PostalCode__s>


<Mailing_Address__StateCode__s>GJ</Mailing_Address__StateCode__s>

<Mailing_Address__CountryCode__s>IN</Mailing_Address__CountryCode__s>


<Mailing_Address__Latitude__s>37.775</Mailing_Address__Latitude__s>

<Mailing_Address__Longitude__s>-122.418</Mailing_Address__Longitude__s>

            </urn:sObjects>
```

```
        </urn:create>
    </soapenv:Body>
</soapenv:Envelope>
```

Update Data Within a Custom Address Field on a Record

This example updates the data stored in the Mailing Address custom address field for record ID `001XXXXXXXXXXXXXXX`.

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:urn="urn:enterprise.soap.sforce.com"
    xmlns:urn1="urn:sobject.enterprise.soap.sforce.com"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Header>
        <urn:SessionHeader>
            <urn:sessionId>$0XXXXXXXXXXXXXXX</urn:sessionId>
        </urn:SessionHeader>
    </soapenv:Header>
    <soapenv:Body>
        <urn:update>
            <urn:sObjects xsi:type="urn1:Account">
                <Id>$001XXXXXXXXXXXXXX</Id>
                <Mailing_Address__Street__s>20 Udhyog
Nagar</Mailing_Address__Street__s>

<Mailing_Address__City__s>Surendranagar</Mailing_Address__City__s>

<Mailing_Address__PostalCode__s>363001</Mailing_Address__PostalCode__s>


<Mailing_Address__StateCode__s>GJ</Mailing_Address__StateCode__s>

<Mailing_Address__CountryCode__s>IN</Mailing_Address__CountryCode__s>


<Mailing_Address__Latitude__s>22.757580</Mailing_Address__Latitude__s>


<Mailing_Address__Longitude__s>71.619350</Mailing_Address__Longitude__s>

            </urn:sObjects>
        </urn:update>
    </soapenv:Body>
</soapenv:Envelope>
```

Delete Data Within a Custom Address Field from a Record

To delete address data stored within a custom address field on a record, update the record. This example deletes the address stored in the Mailing Address custom address field on the Account with record with ID `001XXXXXXXXXXXXXX`.

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:urn="urn:enterprise.soap.sforce.com"
    xmlns:urn1="urn:sobject.enterprise.soap.sforce.com"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Header>
        <urn:SessionHeader>
            <urn:sessionId>$0XXXXXXXXXXXXXXX</urn:sessionId>
        </urn:SessionHeader>
    </soapenv:Header>
    <soapenv:Body>
        <urn:update>
            <urn:sObjects xsi:type="urn1:Account">
                <Id>$001XXXXXXXXXXXXXX</Id>
                <Name>Acc updated</Name>

<urn:fieldsToNull>Mailing_Address__Street__s</urn:fieldsToNull>

<urn:fieldsToNull>Mailing_Address__City__s</urn:fieldsToNull>

<urn:fieldsToNull>Mailing_Address__PostalCode__s</urn:fieldsToNull>

<urn:fieldsToNull>Mailing_Address__StateCode__s</urn:fieldsToNull>

<urn:fieldsToNull>Mailing_Address__CountryCode__s</urn:fieldsToNull>


<urn:fieldsToNull>Mailing_Address__Latitude__s</urn:fieldsToNull>

<urn:fieldsToNull>Mailing_Address__Longitude__s</urn:fieldsToNull>
            </urn:sObjects>
        </urn:update>
    </soapenv:Body>
</soapenv:Envelope>
```

Delete a Record That Contains Data in a Custom Address Field

This example deletes the Account record with ID `001XXXXXXXXXXXXXX`. When a record is deleted, all data for that record is deleted, including the custom address field information.

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:urn="urn:enterprise.soap.sforce.com"
    xmlns:urn1="urn:sobject.enterprise.soap.sforce.com"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Header>
        <urn:SessionHeader>
           <urn:sessionId>$0XXXXXXXXXXXXXXX</urn:sessionId>
        </urn:SessionHeader>
  </soapenv:Header>
  <soapenv:Body>
      <urn:delete>
          <Id>$001XXXXXXXXXXXXXX</Id>
      </urn:delete>
   </soapenv:Body>
</soapenv:Envelope>
```

CHAPTER 10 Tooling API Examples

To retrieve information about fields created with Custom Address Fields, such as the developer name, use Tooling API.

Note: Before you enable custom address fields, review the Custom Address Fields Requirements and Limitations. To discuss the feature and ask questions, join the Custom Address Fields Discussion group on the Trailblazer Community.

Get Information About a Custom Address Field on a Record

This example uses the CustomField REST HTTP GET method to retrieve information about the Mailing_Address__c custom address field, with the CustomField ID `00NXXXXXXXXXXXXXXX`.

HTTP GET example to retrieve information about a custom address field on a record

```
curl
https://MyDomainName.my.salesforce.com/services/data/68.0/tooling/sobjects/CustomField/00NXXXXXXXXXXXXXXX
 -H "Authorization: Bearer token
```

Example response

```
{
  "attributes": {
    "type": "CustomField",
    "url":
"https://MyDomainName.my.salesforce.com/services/dataa68.0/tooling/sobjects/CustomField/00NXXXXXXXXXXXXXXX"

  },
  "Id": "00NXXXXXXXXXXXXXXX",
  "TableEnumOrId": "Account",
  "DeveloperName": "caf",
  "Description": null,
  "Length": null,
  "Precision": 18,
  "Scale": 15,
  "RelationshipLabel": null,
  "SummaryOperation": null,
  "InlineHelpText": null,
  "MaskType": null,
  "MaskChar": null,
  "NamespacePrefix": null,
  "ManageableState": "unmanaged",
  "CreatedDate": "2021-04-07T06:57:22.000+0000",
  "CreatedById": "005XXXXXXXXXXXXXXX",
  "LastModifiedDate": "2021-04-07T06:57:22.000+0000",
  "LastModifiedById": "005XXXXXXXXXXXXXXX",
  "EntityDefinitionId": "Account",
```

```
"Metadata": {
  "businessOwnerGroup": null,
  "businessOwnerUser": null,
  "businessStatus": null,
  "caseSensitive": null,
  "complianceGroup": null,
  "customDataType": null,
  "defaultValue": null,
  "deleteConstraint": null,
  "deprecated": null,
  "description": null,
  "displayFormat": null,
  "displayLocationInDecimal": null,
  "encryptionScheme": null,
  "escapeMarkup": null,
  "externalDeveloperName": null,
  "externalId": false,
  "formula": null,
  "formulaTreatBlanksAs": null,
  "inlineHelpText": null,
  "isAIPredictionField": null,
  "isConvertLeadDisabled": null,
  "isFilteringDisabled": null,
  "isNameField": null,
  "isSortingDisabled": null,
  "label": "caf",
  "length": null,
  "lookupFilter": null,
  "maskChar": null,
  "maskType": null,
  "metadataRelationshipControllingField": null,
  "mktDataLakeFieldAttributes": null,
  "mktDataModelFieldAttributes": null,
  "populateExistingRows": null,
  "precision": null,
  "readOnlyProxy": null,
  "referenceTargetField": null,
  "referenceTo": null,
  "relationshipLabel": null,
  "relationshipName": null,
  "relationshipOrder": null,
  "reparentableMasterDetail": null,
  "required": null,
  "restrictedAdminField": null,
  "scale": null,
  "securityClassification": null,
  "startingNumber": null,
  "stripMarkup": null,
  "summarizedField": null,
  "summaryFilterItems": null,
  "summaryForeignKey": null,
  "summaryOperation": null,
  "trackFeedHistory": false,
  "trackHistory": null,
```

```
    "trackTrending": null,
    "translateData": null,
    "type": "Address",
    "unique": null,
    "urls": null,
    "valueSet": null,
    "visibleLines": null,
    "writeRequiresMasterRead": null
  },
  "FullName": "Account.caf__c"
}
```

Query Information About a Custom Address Field on a Record

This example uses the CustomField REST HTTP Query method to retrieve the developer name of the Mailing_Address__c custom address field with CustomField ID `00NXXXXXXXXXXXXXXX`.

Example query to retrieve the DeveloperName for a custom address field

```
Select+id,DeveloperName+from+CustomField+where+Id='00NXXXXXXXXXXXXXXX'
```

HTTP Query example to retrieve information using the query

```
curl
https://MyDomainName.salesforce.com/services/data/v54.0/tooling/query?q=Select+id,DeveloperName+from+CustomField+where+Id='00NXXXXXXXXXX
 -H "Authorization: Bearer token"
```

Example response

```
{
  "size": 1,
  "totalSize": 1,
  "done": true,
  "queryLocator": null,
  "entityTypeName": "CustomField",
  "records": [
    {
      "attributes": {
        "type": "CustomField",
        "url":
"/services/data/v54.0/tooling/sobjects/CustomField/00NXXXXXXXXXXXXXXX"

      },
      "Id": "00NXXXXXXXXXXXXXXX",
      "DeveloperName": "Mailing_Address"
    }
  ]
}
```