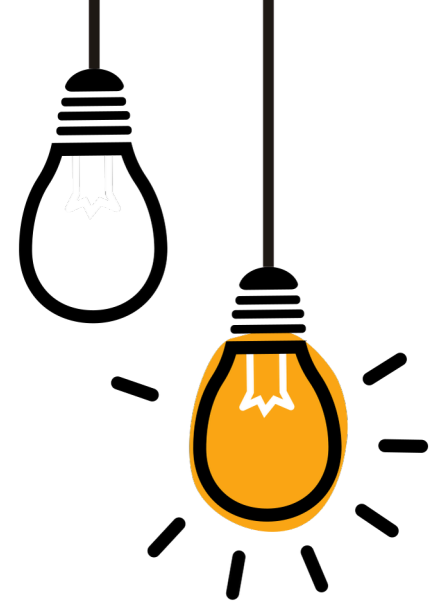
TRAILHEAD IQ



CREATING RECORDS IN LWC WITHOUT USING APEX

Swipe next →

@TrailheadIQ

Creating Records in Salesforce Using LWC createRecord

In LWC, the createRecord method from the lightning/uiRecordApi is a simple and effective way to create new Salesforce records without writing Apex. It handles field-level security and permissions, making it a developer-friendly solution.

Key Features:

- No Apex code required for creating records.
- Automatically respects Salesforce sharing rules and security.
- Easy to integrate into custom Lightning components.

@TrailheadIQ

Swipe next ⟶

When to Use createRecord

Use createRecord when:

- You need to create a new record in Salesforce directly from the frontend.
- You want to leverage the benefits of Lightning Data Service (LDS), such as caching and handling security automatically.

Tip: This is great for use cases like creating new contacts, accounts, or custom object records from a custom UI.

Swipe next ⟶

@TrailheadIQ

Importing createRecord from Lightning UI API

To start using createRecord, you need to import it from the lightning/uiRecordApi module.

```
import { createRecord } from 'lightning/uiRecordApi';
```

Swipe next →

@TrailheadIQ

Basic Structure of createRecord

The createRecord method requires:

- apiName: The object type (e.g., Account, Contact).
- fields: An object representing the field names and values.

```
import { LightningElement } from 'lwc';
import { createRecord } from 'lightning/uiRecordApi';
import ACCOUNT_OBJECT from '@salesforce/schema/Account';
import NAME_FIELD from '@salesforce/schema/Account.Name';

export default class CreateRecordExample extends LightningElement {
    accountName = 'New Account'; // Example field value

    createNewAccount() {
        const fields = {};
        fields[NAME_FIELD.fieldApiName] = this.accountName;

        const recordInput = { apiName: ACCOUNT_OBJECT.objectApiName, fields };

        createRecord(recordInput)
            .then((account) => {
                console.log('Account created with Id: ' + account.id);
            })
            .catch((error) => {
                console.error('Error creating account: ' + error.body.message);
            });
    }
}
```

@TrailheadIQ

Swipe next ⟶

How the Code Works

- **apiName:** Specifies the Salesforce object type (Account, Contact, etc.).

- **fields:** An object where the field names (like Account.Name) are keys, and their values (e.g., 'New Account') are values.

- **createRecord():** This method creates the record and returns the new record’s ID.

Swipe next ⟶

@TrailheadIQ

Creating the Record from a Form

Often, you'll create records based on user input from a form. Here's how you can bind form values to the createRecord method.

HTML

```
<template>
    <lightning-card title="Create Account">
        <lightning-input
            label="Account Name"
            value={accountName}
            onchange={handleInputChange}
        ></lightning-input>
        <lightning-button label="Create Account" onclick={createNewAccount}>
</lightning-button>
    </lightning-card>
</template>
```

Swipe next →

@TrailheadIQ

TRAILHEAD IQ

Js

```
export default class CreateRecordExample extends LightningElement {
    accountName = '';

    handleInputChange(event) {
        this.accountName = event.target.value;
    }

    createNewAccount() {
        const fields = { Name: this.accountName };

        const recordInput = { apiName: 'Account', fields };

        createRecord(recordInput)
            .then((account) => {
                console.log('Account created with Id: ' + account.id);
            })
            .catch((error) => {
                console.error('Error creating account: ' + error.body.message);
            });
    }
}
```

Swipe next ⟶

@TrailheadIQ

Handling Success and Errors

When creating a record, you should handle success and error scenarios properly.

- On Success: Display a success message or redirect the user.
- On Error: Show an error message explaining the issue (e.g., missing required fields).

Here's how you could handle this in the UI:

HTML

```
createRecord(recordInput)
    .then((account) => {
        this.showToast('Success', 'Account created!', 'success');
    })
    .catch((error) => {
        this.showToast('Error', error.body.message, 'error');
    });
    showToast(title, message, variant) {
        const event = new ShowToastEvent({
            title,
            message,
            variant,
         });
        this.dispatchEvent(event);
    }
```

@TrailheadIQ

Swipe next ⟶

TRAILHEAD IQ

Creating Records with Multiple Fields

You can easily create records with multiple fields. Here's how you would add fields like Account.Phone and Account.Industry.

```
const fields = {
    Name: this.accountName,
    Phone: this.accountPhone,
    Industry: this.accountIndustry
};

const recordInput = { apiName: 'Account', fields };

createRecord(recordInput)
    .then((account) => {
        console.log('Account created with Id: ' + account.id);
    })
    .catch((error) => {
        console.error('Error creating account: ' + error.body.message);
    });
```

@TrailheadIQ

Swipe next ⟶

Limitations of createRecord

While createRecord is very powerful, there are some limitations:

- You cannot create child records or related records directly.
- Complex business logic or validations may still require Apex.
- You need to ensure that you provide all required fields, as missing values will result in an error.

Swipe next ⟶

@TrailheadIQ

TRAILHEAD IQ

Summary

The createRecord method in LWC simplifies the process of creating records in Salesforce without writing Apex. It:

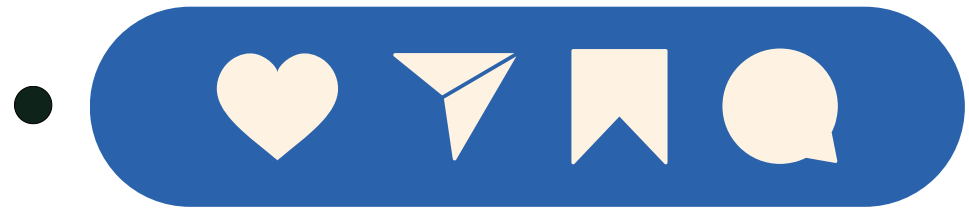
- Automates security and permissions handling.
- Is perfect for creating simple records directly from the frontend.
- Works with form inputs to create dynamic records.
- Use createRecord for seamless and efficient record creation in your custom LWC components!

@TrailheadIQ

Swipe next ⟶

THANK YOU

Remember, every like, share, and comment helps us reach more people and make a bigger impact. Together, we can make the Salesforce community stronger and more informed.#TrailheadIQ#SalesforceCommunity



www.trailheadiq.com

TRAILHEAD IQ