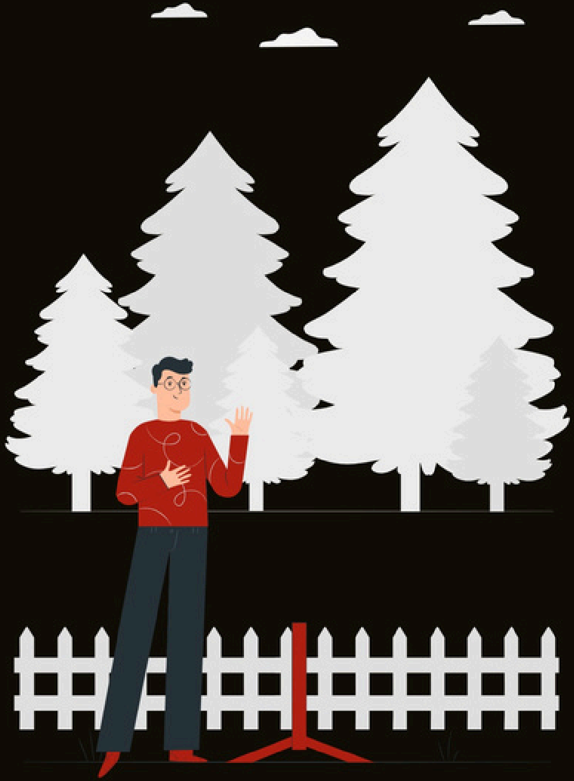
Ext JS Event Handling

- This presentation delves into the intricacies of Ext JS event handling, providing developers with comprehensive strategies to manage events efficiently, thereby enhancing application performance and user experience
- Ext JS Event System: Ext JS provides a powerful event-handling system that allows components to respond to user interactions and system events.
- Event Registration: Use .on() to attach event listeners and .un() to remove them from components.
- Observable Pattern: Classes in Ext JS extend Ext.util.Observable, enabling event-driven programming.

Presented To

Sharath Kumar



Sencha
An Idera, Inc. Company

Presented By

Sushant Kumar Kapri
Satyam Nayak
Anunit Raj
Rehan Rubin Yusuf Khan
Sohaib Raza

EVENT HANDLING

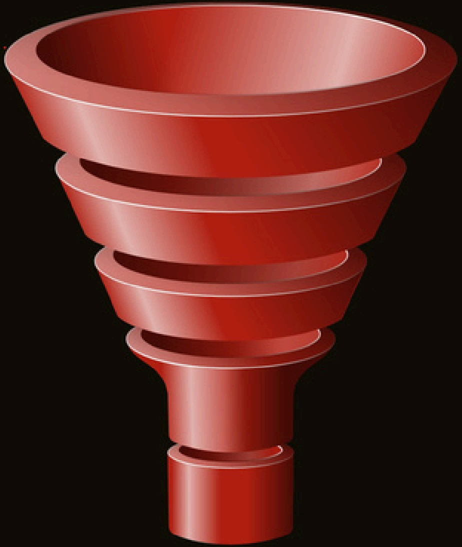
Ext JS Event Handling

Relay Events: Allows one component to listen and re-fire another component's events.

Best Practices: Always clean up event listeners to prevent memory leaks and improve application performance.

Understanding Events in Ext JS

Key Concepts and Definitions



1 What are Events?

Events are occurrences that take place in the browser, vital for user interactions.

2 Importance of Events in Ext JS

Events facilitate application behavior and user engagement through interactions.

3 Event-driven Architecture

An architecture model where the application responds to events dynamically.

4 Types of Events

Includes clicks, keyboard inputs, and mouse movements, crucial for application interactions.

5 Event Propagation

Understanding how events bubble up or trickle down through the DOM is essential.

Understanding Event Listeners

Key Concepts and Examples

Definition of Event Listeners

Event listeners are functions that trigger actions in response to specific events.

Purpose of Event Listeners

They define how applications react when certain events occur, improving interactivity.

Example Usage

For instance, using 'addEventListener' for click events to execute functions.

DOM Elements Attachment

Event listeners can be attached to various DOM elements, enhancing user interaction.

Separation of Concerns

They allow developers to keep event logic distinct from the main application code.

Adding Event Listeners in Ext JS

Effective Techniques for Event Handling

1 **Using the `on` method**

You can add event listeners using the `on` method. Example: Ext.get('myElement').on('click', function() { console.log('Element clicked'); });

2 **Component Configuration**

Event listeners can also be added through component configuration. Example: Ext.create('Ext.button.Button', { text: 'Click Me', handler: function() { alert('Button clicked'); }});

3 **Using Animations**

Enhance user feedback with animations when events are triggered, improving user experience.

Understanding the Event Object in JavaScript

Key Attributes and Functions

Event Object Creation

An event object is generated when an event occurs, encapsulating event details.

Type Attribute

The 'type' property indicates the event type, such as 'click'.

Target Element

The 'target' property identifies the element that triggered the event.

Current Target

The 'currentTarget' property points to the element with the event listener attached.

Prevent Default Action

The 'preventDefault()' method stops the default behavior of the event.

Example Usage

Example code shows how to access event properties in a click event.

Enhancing User Experience with Animation

Using animation to enhance user interactions effectively

Animation improves interaction

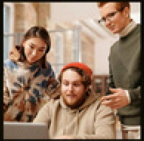
Animations can significantly enhance user interaction by providing visual feedback.

Visual appeal

Creates a visually appealing experience that attracts users' attention.

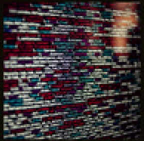
Immediate feedback

Provides immediate feedback to user actions, making the interface intuitive.



Increased engagement

Increases engagement and retention, encouraging users to stay longer.



Example of animation

Example code to highlight an element upon click enhances interactivity.

Handling Click Events in JavaScript

A Real-World Example

- **Scenario Overview**
  Updating a message when a button is clicked enhances user interaction.
- **Code Implementation**
  The provided JavaScript code creates a button with a click event handler.
- **Alert Message**
  Upon clicking, an alert displays the message 'Button has been clicked.'
- **User Experience Improvement**
  Using animations like fade-in or slide effects can enhance the visual appeal.

Best Practices for Event Handling

Key Practices for Event Handlers

🟥	2.02K	Keep it Simple
🟫	2.02K	Use Namespaces
🟧	2.02K	Remove Listeners When Not Needed
🟦	2.02K	Debounce or Throttle Events
🟥	2.02K	Conclusion

High-quality Design

Make your presentations stand out, by letting Presentations.AI do the design work for you - in seconds.

Amazing Templates

Save time and effort with ready-made designs that are fully customisable.

Easy Collaboration

Prepare for the unexpected and delight your audience with a seamless presentation that is always right on time.

Works Everywhere

Powerful collaboration tools that help you create, organise, review and publish your content to any screen.

Summary of Ext JS Event Handling

Essential Insights for Effective Ext JS Applications

1. **Vital Role of Events**
   Events are crucial in enabling user interactions within Ext JS applications, making them responsive.

2. **Understanding Event Types**
   Grasping the various event types alongside the event object is fundamental for effective programming.

3. **Managing Event Listeners**
   Efficiently adding and managing event listeners is key to optimizing application performance.

4. **Enhancing User Experience**
   Utilizing animations can significantly improve the overall user experience in applications.

5. **Best Practices in Coding**
   Adhering to best practices fosters clean, efficient, and maintainable code throughout development.

Thank You