Explaining Visual Changes in Web Interfaces

Brian Burg, Andrew J Ko, Michael Ernst

University of Washington
About

The ACM Symposium on User Interface Software and Technology (UIST) is the premier forum for innovations in human-computer interfaces. Sponsored by ACM special interest groups on computer-human interaction (SIGCHI) and computer graphics (SIGGRAPH), UIST brings together people from diverse areas including graphical & web user interfaces, tangible & ubiquitous computing, virtual & augmented reality, multimedia, new input & output devices, and CSCW. The intimate size and intensive program make UIST an ideal opportunity to exchange research results and ideas. Join us in Charlotte!
Feature Location

Feature location is the activity of identifying an initial location in the source code that implements functionality in a software system.

Feature Location
for Interactive Web Content

Apple Watch Gallery
Browse all models

Authors ▼
How can I interact with it?
How is the visual effect achieved?

DOM & CSS  ➔  Visual Output
What code is ultimately responsible?
No Links Between Source and State  
Scattered State  
No Output History
Scry

Staged, Interactive Feature Location
Staged, Interactive Feature Location

Output Examples

[Authors ▼

Authors ▼]
Staged, Interactive Feature Location
Staged, Interactive Feature Location

Output Examples

State Differences

JavaScript Mutations
Authors ▼

Δ(pre, post)

Output Examples  State Differences  JavaScript Mutations
What determines visual appearance?

DOM Tree Structure

CSS Style Properties
Color, layout mode, visual styling, text rendering, handling of children

Rendering Engine

Visual State
Target Element Snapshots

DOM Subtree

Computed Styles & Related Rules

Element Screenshots

Mutation Operations
Why does visual appearance change?

DOM Tree Mutations

- Insertion
- Deletion
- Ordinal Change
- Attributes & States

Style Property Changes

- Rule Changes
  - Inline Styles
  - Animations
Detecting changes in appearance

Painted Rects

Do they intersect? → Visual Diff >1%? → Commit New Snapshot

- NO → NO → NO
- YES → YES
Comparing State Snapshots

Per-element change summaries
Structure: Insertion, Deletion, Attributes, …
Styles: Added, Removed, Value Change

Relies on stable DOM element identity
Doesn’t work well when view state is split from DOM
Change-Relevant Operation Slicing

1. Instrument and record mutation operations.

2. Build a dependency graph for operations between the pre-state and post-state.

3. Based on change summary, find an equivalent mutation operation to explain the change.

4. Return equivalent operation + dependencies
Technical Challenges

Visual containment, stacking

Software vs hardware rendering

Unstable DOM element identities

Megamorphic call sites in library code

Pruning ineffective styles and attributes
Summary

Feature Location via Visual States
States can be automatically captured when drawing.

Juxtapose Captured Inputs and Outputs
State and output snapshots help explain each other.

Diff Markers Filter Relevant Operations
Slicing algorithms can show responsible operations.
Explaining Visual Changes in Web Interfaces

Brian Burg, Andrew J Ko, Michael Ernst

University of Washington