Guiding Web Inspection with Tutorial Keyword Frequency

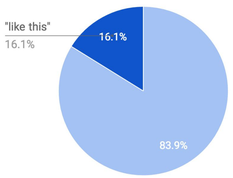
Sarah Lim

Northwestern University, Delta Lab

Motivation

Many self-taught front-end programmers build their skills in an **ad-hoc, feature-centric manner**.

Features are an especially meaningful vehicle for understanding among novices who **lack technical vocabulary** to describe concepts more formally.

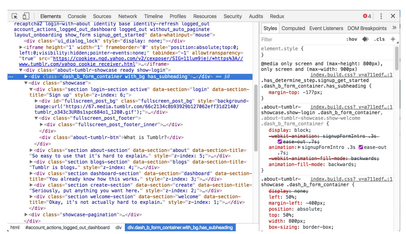


16% *of StackOverflow questions contain the phrase* ***"like this"*** *in the question title*

How might we leverage feature-rich professional websites as ***contextual worked examples*** *for end-user programmers?*

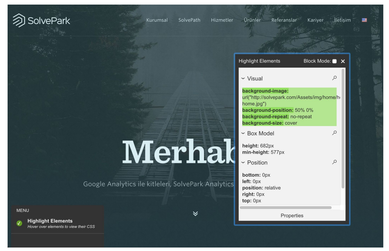
Worked examples have been shown to effectively increase **near transfer rates** during instructional tasks.

Unfortunately, professional website source code is **available but not accessible**. Existing tools are designed for professionals, imposing significant **extraneous cognitive load** on novices.



Contribution

Our system is a **scaffolding feature inspector** designed to help end-users detect and reverse engineer features on professional websites.



System uses heuristics to **organize**, **emphasize, and de-emphasize information** based on relevance to the feature of interest.

1. Functional Grouping of Source Code Units

Property Type	Examples
Position	`position: absolute;` `top: 0;`
Box Model	`margin: 0 auto;` `box-sizing: border-box;`
Visual	`color: #e4e4e4;` `font-family: Georgia, serif;`

2. Crowdsourcing Prioritization from Existing Tutorials

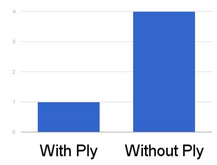
In order to determine the relevance of a line of code in a professional example, we crowdsource **existing tutorials** for features and **pre-compute** the frequency of particular source code units (e.g. CSS properties, DOM class name patterns).

Preliminary Results

In pilot testing, **8 front-end developers** (6 novice, 2 experienced) attempted to reverse engineer features on professional websites.

All 6 novices had **difficulty locating the correct DOM nodes for inspection** *(structural design patterns)*.

Once the correct nodes were located, **4 out of 6 identified critical CSS properties** within the allotted 15 minutes using the tool, compared to **1 out of 6** without the tool *(syntactical design patterns)*.



4 out of 6 *users correctly identified critical CSS properties on professional examples using our system*

All users described the system as "**much easier to use**" and "**less intimidating**" than the standard Chrome Developer Tools inspector.

Future Work

Extending prioritization affordances to DOM nodes to help learners understand how features are structured in the DOM prior to CSS styling

Applying unsupervised learning techniques to crowdsourced tutorials and examples, in order to develop more sophisticated predictive models

Supporting "guess and check" behavior frequently exhibited by novices and end-user programmers