Quid: A web-based DSL for defining User Interfaces applied to Web Components

Pedro J. Molina, PhD. `pjmolina@metadev.pro`

Metadev S.L. Seville, Spain

Abstract. User Interface construction is a recurrent topic in Software Engineering: multiples tools ranging from textual, graphical design tools exists to help in this task. On the other hand, the fast pace of front-end industrial frameworks makes such editors tools obsolete as soon as new technology emerges. The work presented here, **Quid**, introduces a web-based DSL with focus on minimal accidental complexity, removing the accessory markup and a WYSIWYG environment to provide real-time feedback to users. Moreover, the UI specification built in this way is platform-independent: its primitives can be extended, and target different platforms: using model transformations and code generation for generating software artifacts like Native Web Components or Angular code.

1 Introduction

Quid is a web tool to define the composition of User Interfaces in an abstract way. It follows a component model based in the *W3C Web components*[4] and the key concept of (Abstract Interaction Objects)[1] for providing platform independence. The tool allows composition of UIs to reuse and create complex designs, allowing the specification of *Conceptual User Interface Patterns*[2].

2 Motivation: Web Components

The W3C Web Component specification[4] is a living specification been standardized in commercial browsers.

The spec once implemented in browsers will simplify application development, in the way it brings a full component model for user interfaces in a native way (implemented by browsers directly and exposing a common API to developers). This will allow the mix of components develop with different languages, or frameworks to compose a web page or application.

This technology is starting to be ready to use on desktop and mobile browsers. However, at the time of writing this, there is a **lack of tooling for composing web components**. Here is where **Quid** is designed to help to compose Web Components.

3 Quid

Quid is a web tool under heavy development. A general overview can be seen on Figure 1. In the following sections, the main features will be described:
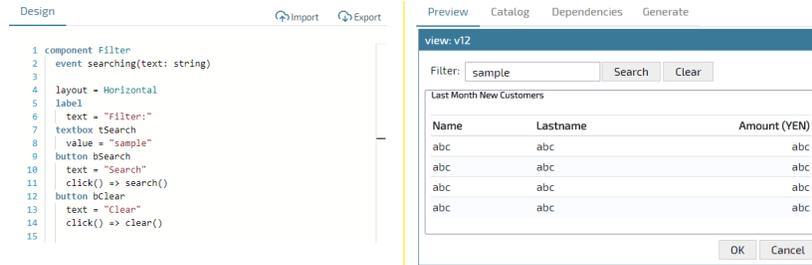

Fig. 1. Quid Overview: design and preview panes

3.1 Quid DSL

Quid provides a textual DSL with minimal markup (brackets and delimiters) to avoid unneeded accidental complexity. Indentation is used to represent **visual containment** relationship. This means controls can be nested just using the `TAB` key and reversed using `Shift-TAB`. The DSL editor is text-based providing colorization and code completion using the keys `Control-Space`

Sample UI specification on Quid:

```
component Filter
    label l1
        text = "Filter:"
    textbox searchText
        tooltip = "type here for search"
    button cmdSearch
     text = "Search"
        (click) => onSearch()


view Page1
    Filter f1
    ...
```

In the sample provided, a component named **Filter** is defined. The indentation suggests the next three components are defined inside the Filter. This containment relationship is both graphical, and semantical (scope and encapsulation). The subcomponents can have properties and can be configured on demand. Finally, an event is captured to trigger an execution after a click event

on a button. The component defined in this way is immediately available for instantiation in the view defined below, where a **Filter** is instantiated with the name of **f1**.

3.2 Tool support

Integrated Validation The DSL provides a real-time integrated validation. On each keystroke, a validation procedure is triggered involving: tokenization and parsing stage, AST construction, semantical validation, and error reporting. If lexical or semantical errors are found, they are provided just in time in the zone below the editor and inline in the editor using colors to indicate the source of the errors in the exacts lines.

Preview Mode Once the validation is successful, the tool provides a Preview window where the actual design is rendered to provide a WYSIWYG experience with immediate feedback to the designer. In this kind of tools, this immediate feedback is the key to enhancing and keeping the productivity of users within the tool. The preview mode allows interaction, depending on the behavior defined for the User Interface.

Catalog Quid is extensible in terms of basic building blocks to be used for design. The catalog (see Figure 2) is the place to explore the available primitives, its properties and events, and to import third-party components when needed. Changing the base building blocks allows Quid to be used in different design scenarios without changing the core tool. Moreover, third-party components can be incorporated to change the palette of building blocks.
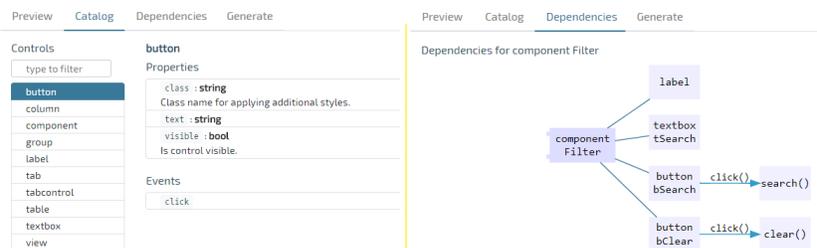

Fig. 2. Quid: Component Catalog and Dependencies View

Code Generation Finally, a code generation pane provides an integrated toolset to translate the specification into different code artifacts depending on the target platform. **Quid** provides code generation to Native Web Components

generating HTML, CSS & Javascript code to allow a running implementation of the spec. As Work in Progress, other target platforms are under consideration such as Angular Elements, Ionic Stencil, Polymer 3, VueJS, React, etc.

3.3 Use Cases

The tool is in its early stages. Nevertheless, it can support the following use cases:

1. Define components and composition of components for fast prototyping of UI (atomic design)
2. Form design
3. Selection of a closed catalog of components (company palette) to enforce common look & feel on designed UI
4. Custom code generation for arbitrary architectures and language stack.
5. Use the Quid interpreter to render the UI on third-party applications.

The tool has been used with success in an industrial international project to design and generate User Interfaces for an Investment Banking Application lead by one of the biggest software consultancy firms in Spain.

4 Conclusions

There are myriad design tools for User Interfaces. The novelty of Quid is to focus on (a) abstract UI representation, (b) providing a web-based DSL (no installation of software is required, bringing a full-featured editor on the web), and (c) early embracing a component model fully compatible with W3C Web Components to help using, importing and combining such web components.

The immediate feedback principle is also a great feature in the tool. This immediate feedback can be slower as the specification grows. To mitigate such problems, techniques are been applied to partition the specification in separate chunks and to render partial views the **Preview** window.

Latest version of **Quid** can be found online at: `https://quid.metadev.pro`[3].

References

1. Bodart. F., Vanderdonckt, J.: Widget Standarization through Abstract Interaction Objects. Proceedings of 1st Int. Conf. of Applied Ergonomics ICAE'96. Pages 300-305. USA Publishing, Istamboul - West Lafayete, 1996.
2. Molina, P.J., Pastor, O. Marti S., Fons, J.J.: Specifying Conceptual User Interface Patterns in an Object-Oriented Method with Code Generation. Proceedings of 2nd IEEE Workshop on User Interface for Data Intensive Systems UIDIS'01. Pages 72-79. IEEE Computer Society, Zurich, Switzerland, 2001. ISBN 0-7695-0834-0.
3. Molina, P.J. Quid `https://quid.metadev.pro` 2018.
4. The World Wide Web Consortium, et al. Web Component Specification Working Drafts, 2014-2018. `https://www.w3.org/standards/techs/components`