Análise de APIs em Javascript para Criação de Interfaces Web Ricas

Vinicius Vielmo Cogo[1], **Vitor C. F. Gomes**[1], **Gustavo Rissetti**[1], **Andrea S. Charão**[1]

[1]PET – Universidade Federal de Santa Maria (UFSM)

{vielmo,vconrado,rissetti,andrea}@inf.ufsm.br

Abstract. *In recent years, there have been great advances in Javascript APIs to develop rich Web interfaces. This article examines a few of these APIs from the point of view of resources that enrich the user experience. For each feature, it discusses the advantages and disadvantages of each selected API, based on a set of criteria. The results of this analysis can help Web application developers to choose and combine APIs to produce friendly user interfaces.*

Resumo. *Nos últimos anos surgiram diversas APIs em linguagem Javascript para desenvolvimento de interfaces Web. Neste artigo, analisa-se algumas dessas APIs sob o ponto de vista de recursos que enriquecem a interação com o usuário. Para cada recurso, discute-se as vantagens e desvantagens de cada API selecionada, com base em um conjunto de critérios. Os resultados dessa análise podem auxiliar os desenvolvedores de aplicações Web a escolher e combinar APIs para produzir interfaces amigáveis.*

1. Introdução

Atualmente existe uma tendência de migração de aplicações do *desktop* para a plataforma Web, em que o navegador é o meio de interação com o usuário [Bozzon et al. 2006]. Seguindo esta tendência, surgiram tecnologias visando enriquecer as interfaces Web, que originalmente eram menos amigáveis que as interfaces para *desktop*. Isso culminou com a criação de novas interfaces de programação (*Application Programming Interfaces* - APIs), contribuindo para o reaproveitamento de código e a propagação de técnicas de usabilidade, trazendo assim benefícios para o programador e para o usuário.

Dentre as APIs disponíveis atualmente, grande parte utiliza a tecnologia Ajax para a linguagem Javascript, que é suportada pela maioria dos navegadores Web modernos. Dada a grande quantidade e diversidade de novas APIs, os desenvolvedores deparam-se com a necessidade de investigá-las e avaliá-las a fim de determinar seus reais benefícios para o desenvolvimento de novas interfaces de aplicações Web. Inserindo-se neste contexto, o presente trabalho realiza uma análise de componentes de interface oferecidos por APIs Javascript populares, investigando também a possibilidade de combiná-las em aplicativos Web. Para isso, este trabalho apresenta primeiramente uma contextualização sobre APIs (seção 2), seguida da descrição da metodologia empregada neste estudo (seção 3). Na sequência (seção 4) são apresentados os componentes de interface e as análises das APIs que os implementam. Por fim, são feitas algumas considerações acerca do trabalho e de sua contribuição para desenvolvedores de aplicações Web mais ricas.

2. APIs em Javascript

Existem diversas APIs em Javascript para o desenvolvimento de interfaces de Web disponíveis na Internet. Também são numerosas as comparações entre APIs encontradas em fóruns de discussão e *sites* especializados. No entanto, essas avaliações geralmente não esclarecem a metodologia utilizada e consideram as APIs como um conjunto fechado de componentes, elegendo uma API como a favorita para o desenvolvimento de determinada aplicação. Este fato indica que não está sendo considerada a utilização conjunta dessas APIs, abordagem esta que permitiria ao desenvolvedor usufruir dos melhores recursos de cada API.

Este trabalho aborda algumas APIs populares considerando cada componente separadamente, com vistas a permitir ao desenvolvedor escolher APIs com base nos componentes que serão utilizados, permitindo um aumento de flexibilidade no desenvolvimento de suas aplicações e contribuindo para o enriquecimento da interface criada. Para isso, foram analisadas as APIs Tabtastic [Kistener 2004], Tipster [Turnbull 2007], WalterZorn [Zorn 2008], Rico [Cowin 2009] e overLib [Bosrup et al. 2009], que disponibilizam suporte a componentes específicos. Além dessas, também foram analisadas APIs que reúnem diversos componentes: Yahoo! User Interface (YUI) [Yahoo! Inc. 2009], MooTools [Proietti 2009], jQuery [Resig 2009], QooXDoo [1&1 Internet AG 2009], MochiKit [Mochi Media, Inc. 2009], script.aculo.us [Fuchs 2009] e Dojo [The Dojo Foundation 2009]. A próxima seção apresenta em detalhes a metodologia utilizada para escolha e análise destas APIs.

3. Metodologia

A forma de trabalho adotada nesta análise baseou-se inicialmente no levantamento dos principais requisitos para APIs voltadas à criação de interfaces Web. Considerando a inviabilidade da abrangência de todas APIs disponíveis atualmente, fez-se necessário definir alguns critérios de seleção. Foram então determinados dois tipos de critérios, os eliminatórios - que poderiam excluir uma determinada API da análise - e os qualitativos - que definiriam as características a serem analisadas nas APIs escolhidas.

Como critérios eliminatórios, considerou-se a disponibilidade de documentação sobre a API na Internet e sua portabilidade quanto aos navegadores, ou seja a API deveria manter suas funcionalidades nos navegadores mais populares entre os usuários da Web (Internet Explorer, Firefox, Opera e Safari). Por sua vez, os critérios qualitativos foram, nesta ordem: qualidade da documentação, facilidade de utilização, licença de distribuição (preferencialmente as livres), popularidade (casos de uso, comunidade e estrutura do código). Além disso, considerou-se outros critérios qualitativos referentes aos componentes internos às APIs: facilidade de utilização, possibilidades de configuração (opções disponíveis para utilização do componente), interface de programação (formas de acesso e interação com o componente) e por último qualidade e flexibilidade visual do componente.

No início do trabalho foram pré-selecionadas dezesseis (16) APIs populares em Javascript. Dentre estas, doze (12) delas passaram pelos critérios eliminatórios, indicando que quatro (4) foram excluídas da análise apresentada neste artigo. Para as APIs escolhidas, citadas na seção anterior, foram analisados dez (10) componentes e recursos de interface de usuário, descritos na próxima seção: abas, *Drag-and-Drop*, *Accordion*, árvores,

Tips, calendário, menu, caixa de diálogo, *Resizable* e *Sortables*. Para a análise, foram criadas pequenas aplicações utilizando os componentes de todas as API selecionadas.

4. Análise dos Componentes de Interface

4.1. Abas

O componente aba permite a organização do conteúdo de uma aplicação em *containers* que podem ser acessados através de abas de navegação. Seu principal objetivo é permitir a criação de um ambiente de navegação mais limpo e organizado.

Para este item foram analisadas as seguintes APIs: Yahoo! User Interface (YUI), MooTools, Tabtastic e jQuery. Apesar de todas as APIs possuírem uma documentação satisfatória para este item, a MooTols se destaca neste quesito. Em contrapartida, esta API não possui suporte nativo a este componente, sendo necessário incluir um *plugin* para a sua utilização. A YUI destaca-se pela organização do código-fonte em subpastas, permitindo a inclusão somente do código necessário para a utilização deste componente. Apesar disso, isso gera a necessidade de um melhor domínio da hierarquia utilizada por esta API. A terceira API, Tabstatic, possui funções extras não disponibilizadas nas demais APIs, confirmando que se trata de uma API especializada neste componente. Além disso, sua utilização é bastante simples. A última API analisada neste quesito, jQuery, também compartilha desta característica, possuindo ainda bons exemplos e excelente documentação, destacando-se em relação às demais. Um exemplo de abas da jQuery pode ser visto na figura 1a.

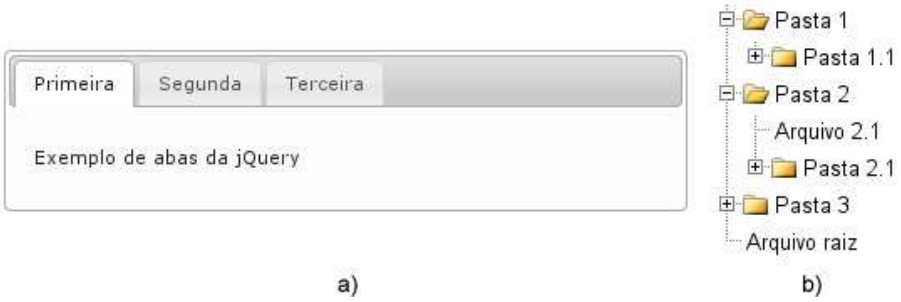

Figura 1. a) Abas da jQuery. b) Árvore da YUI

4.2. Drag-and-Drop

O objeto de estudo *Drag-and-Drop* não é um componente propriamente dito, mas sim um recurso que pode ser inserido em alguns objetos da interface. Ele consiste na possibilidade do usuário arrastar e soltar elementos ao longo da interface Web, reorganizando assim a disposição do conteúdo de tal forma que se crie um ambiente personalizado.

Uma grande utilização da característica *Drag-and-Drop* é em carrinhos de compras em *sites* de comércio eletrônico, em que o usuário literalmente arrasta os produtos para o seu carrinho de compras. Também é utilizado em jogos baseados na Web, em que se faz necessário arrastar componentes pela tela e largá-los em outra parte da mesma.

Para este item, foram selecionadas as seguintes APIs: jQuery, Mochikit, MooTools, Rico, script.aculo.us e YUI. Todas obtiveram um desempenho satisfatório na maioria dos critérios qualitativos. Dentre as poucas diferenças existentes entre as APIs, neste

tópico, encontra-se a profundidade dos exemplos adotados e da documentação disponível, sendo que os componentes de *Drag-and-Drop* das APIs citadas neste item cumprem os requisitos básicos necessários para uma boa interface com o usuário. Um dos poucos recursos exclusivos de uma API é da script.aculo.us, que possui uma propriedade chamada *scroll*, que possibilita o usuário arrastar um objeto por toda a tela, independentemente se esta possui barra de rolagem ou não. Outros recursos complementares que são encontrados na maioria das APIs são: a possibilidade de limitar a região de arrasto, tratamento de colisões com alvos para o momento em que o usuário solta o elemento, escolha da região do elemento possibilitará o arrastar e soltar, além de permitirem o uso ou não alterações das características do componente em tempo de execução.

4.3. Accordion

Accordion é um componente que permite a apresentação de vários tópicos de conteúdos em um pequeno espaço. Ele apresenta todos os títulos dos tópicos, mas somente o texto de descrição de um tópico escolhido. Ao escolher-se outro título, o *Accordion* fecha a descrição corrente e abre a nova descrição selecionada. Com isso, pode-se obter diversos textos em um pequeno espaço e de maneira organizada.

Para este componente, passaram na seleção as APIs jQuery, MooTools e Rico. Quanto à facilidade de uso, considerou-se, nesta análise, a API Rico como sendo a mais indicada, mesmo esta possuindo documentação apenas internamente ao exemplo. A MooTools e a jQuery possuem boa documentação, exemplos claros e simples, assim como um bom desempenho e aparência.

Pode-se destacar algumas propriedades desejáveis que foram encontradas nas três (3) APIs selecionadas para este item, sendo elas: animação na troca de página, boa integração com folhas de estilo em cascata e espaço reservado para títulos e conteúdos separadamente. Um *Accordion* implementado usando a API Rico pode ser visto na figura 2a.

4.4. Árvores

O componente Árvore permite a organização e visualização de dados estruturados hierarquicamente, semelhante aos sistemas de arquivos de sistemas operacionais, que permitem ao usuário expandir ou retrair um item ou sub-item do componente. Através da utilização de itens e subitens aninhados é possível organizar estruturas hierárquicas como sistemas de arquivos, menus, árvores genealógicas entre outros usos.

As APIs que foram selecionadas para este item foram: YUI, MooTools e QooXDoo. A YUI se destaca em relação às outras duas, principalmente no quesito de documentação. Por sua vez, a MooTools possui um *plugin* que implementa este componente, não fazendo parte do código nativo da API. A QooXDoo possui difícil utilização frente às demais. Dentre alguns requisitos desejáveis para este componente, encontram-se: utilização de um arquivo XML como base para os dados, navegação simulando itens e subitens, animação na navegação, criação e remoção de nós em tempo de execução, entre outros, que foram satisfatoriamente cumpridos pelas três (3) APIs. Um exemplo de árvore usando a API da YUI pode ser visto na figura 1b.

4.5. Tips

Para uma maior interatividade de navegação em um ambiente Web, pode-se utilizar o recurso de *Tips*. Este recurso consiste, geralmente, em informações adicionais a respeito de alguma funcionalidade da aplicação, mostrada quando o mouse é passado sobre uma região pré-definida. Um texto com seu *Tip* associado é mostrado na figura 2b.

Para analisar tal componente, foram selecionadas as seguintes APIs: MooTools, overLib, Tipster, WalterZorn e QooXDoo. Dentre estas, as APIs MooTools e overLib se destacam por serem as únicas que implementam *Tips* que permitem o uso de folhas de estilo em cascata (CSS) para o elemento descritivo. Todas as APIs supriram os requisitos básicos para o uso de *Tips*, que são: detecção de eventos diferentes sobre os elementos aos quais as *Tips* estão ligadas, ter boa documentação, possuir exemplos claros, ser aplicado em imagens (exceto overLib) e possuir animações para o elemento descritivo.

4.6. Calendário

O componente calendário é responsável por exibir uma estrutura tabular com os dias dos meses organizados semanalmente. Este item possui como principal funcionalidade facilitar a apresentação e seleção de datas, principalmente em formulários. Além disso, com o uso deste componente é possível diminuir os problemas relacionados aos erros gerados pela falta de padronização da expressão que representa uma data.

Foram selecionadas três (3) APIs, sendo elas: jQuery, QooXDoo e YUI. Dentre as principais necessidades de uma interface quanto ao uso de calendários, encontram-se: seleção de um dia específico, relação com *input texts*, desabilitar a seleção de alguns dias (nos casos de feriados por exemplo), seleção de um intervalo de tempo, entre outros. Por motivos de facilidade de uso e flexibilidade de configuração, a YUI foi selecionada como a mais indicada para uso em interfaces de usuário. Por sua vez, a QooXDoo é contra-indicada quando deseja-se usar somente este componente da API, pois está atrelado a ele uma grande quantidade de classes, diferentemente do que ocorre com outras APIs que suportam este componente.

4.7. Menu

O componente de menu é um dos mais importantes para uma boa interface de usuário. Assim como pode facilitar muito o acesso às informações, pode também dificultar a usabilidade da interface. Existem diversas formas de menus que podem ser utilizadas, como por exemplo: menus primários, menu *drop-down*, menu olho-de-peixe, dentre outros. Trata-se de um importante recurso para aproximar interfaces Web das interfaces existentes em aplicações para *desktop*.

Para este item foram selecionadas as seguintes APIs: YUI, MooTools e Dojo Toolkit. Dentre elas, a API mais indicada para o uso é a YUI, pois possui fácil utilização, boa documentação e diversas formas de uso. Enquanto a Dojo ToolKit apenas implementa um modelo de menu, conhecido como olho-de-peixe, a MooTools possui difícil utilização e efeitos de animação.

4.8. Caixa de Diálogo

Uma caixa de diálogo é um componente que permite comunicar e dialogar com os usuários sem criar novas janelas ou abrir novas abas no navegador, podendo ser aplicado como avisos ou notícias de última hora, por exemplo.

Neste item foram selecionadas as APIs YUI e jQuery. A YUI possui boa documentação, porém possui poucos exemplos, considerados fracos entre as possibilidades. Este item está subdividido em SimpleDialog, que são os diálogos mais simples, como sim/não, ok/cancel, e em Dialog, que são formas mais complexas e dinâmicas de interação. O componente da jQuery é de fácil utilização, possuindo uma boa documentação e apresentando exemplos claros. Esta última foi portanto considerada a melhor API a ser usada para o item Caixa de Diálogo, por ser mais bem documentada e exemplificada.

4.9. Resizable

Assim como o *Drag-and-Drop*, o *Resizable* é uma propriedade ou característica que pode ser aplicada a algum componente dentro de uma interface de usuário. Esta propriedade permite ao usuário redimensionar componentes da interface, personalizando assim a disposição de conteúdo na tela. Ela também é utilizada em *startpages* e interfaces que simulem aplicativos *desktop*, uma tendência cada vez maior. Com esta propriedade também é possível apresentar um determinado conteúdo em tela cheia, aumentando assim o aproveitamento da tela enquanto se navega pela Internet. Um exemplo da propriedade *Resizable* pode ser visto na figura 2c.

As APIs que foram selecionadas para este item foram: YUI, MooTools, jQuery, QooXDoo. A YUI apresenta uma boa documentação sobre este componente. É simples, mas requer a escrita de uma quantidade relativamente grande de código, dificultando algumas vezes a programação. A MooTools apresenta uma fácil utilização e boa documentação no *site* oficial, com exemplos simples e claros. O resultado final tem uma boa aparência. A API jQuery também apresenta uma boa documentação e é de fácil utilização. Ela se destaca em relação à MooTools pela grande quantidade de opções que apresenta em relação ao item *Resizable*. Já a API QooXDoo apresenta exemplos muitos simples, e não se destaca em relação às outras, não sendo uma boa escolha de API a ser utilizada.

Pode-se assim concluir que a API mais indicada a ser usada para o item *Resizable* é a jQuery, pois apresenta mais opções que as demais e possui melhor resultado final.

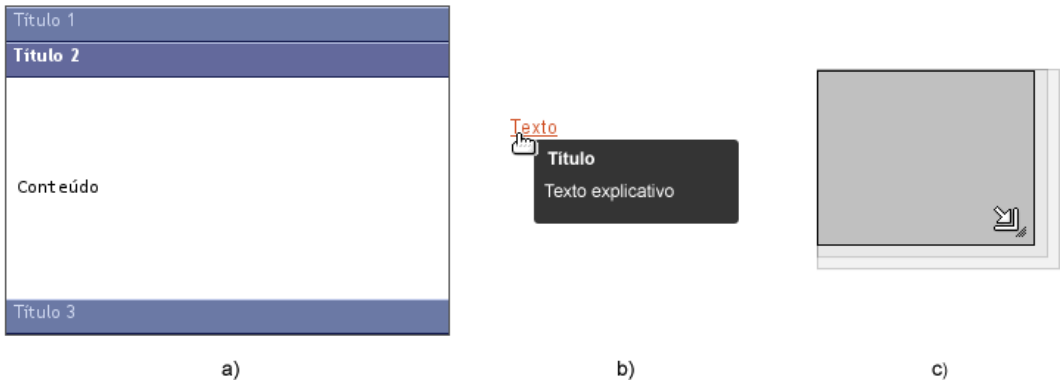

Figura 2. a) *Accordion* da Rico. b) Tips da MooTools. c) Resizable da jQuery

4.10. Sortables

Existem aplicações Web em que é necessário visualizar e manipular uma tabela ordenada por algum campo específico. O recurso *Sortables* visa suprir esta necessidade, possibili-

tando a criação de tabelas que podem ser ordenadas por qualquer coluna, para aumentar a facilidade de obtenção das informações desejadas.

O componente da API da YUI não foi comparado com as outras, pois é implementada através de uma propriedade vinculada a uma tabela, transformando-as em uma Sortable, sendo de fácil utilização e poucas opções avançadas. A jQuery possui uma boa documentação neste item, com exemplos claros e objetivos e fácil utilização. A script.aculo.us possui excelentes exemplos de uso, porém a documentação deixa a desejar por não tratar todas as possibilidades do item. Esta avaliação indica o componente da jQuery como o melhor nesta categoria.

4.11. Síntese dos Resultados

Para facilitar a visualização das APIs selecionadas para cada componente, a tabela 1 apresenta uma síntese desta avaliação.

Tabela 1. APIs indicadas para cada componente

Componente	API
Abas	jQuery
Drag and Drop	jQuery, Mochikit, MooTools, Rico, script.aculo.us e YUI
Accordion	Rico
Árvores	YUI
Tips	MooTools e overLib
Calendário	YUI
Menu	YUI
Caixa de Diálogo	jQuery
Resizable	jQuery
Sortables	jQuery

5. Considerações Finais

Este trabalho apresentou uma análise das principais APIs em linguagem Javascript para a criação de interfaces Web. A avaliação foi feita através da análise dos componentes internos das APIs. Foi possível observar que não existe uma única API que se destaca em todos os itens, indicando a importância de integrar as APIs para o uso dos componentes que se sobressaem em cada quesito. De fato, estima-se que com a utilização integrada das APIs seja mais fácil desenvolver aplicações Web com características de aplicações *desktop*, sem redução de interatividade com o usuário.

Apesar de não existir um predomínio de uma API, observou-se que a YUI possui uma documentação muito satisfatória para todos os itens analisados. Destacam-se também os exemplos disponibilizados pela jQuery, que na maioria das vezes são simples e objetivos.

Referências

1&1 Internet AG (2009). *Qooxdoo*. Disponível em http://qooxdoo.org/, Acesso em 18/04/2009.

Bosrup, E., Steinman, D., Hirsch, F., Mikkonen, T. J., McKenas, A., Arzu, F., and Delbecq, D. (2009). *overLib*. Disponível em http://www.bosrup.com/web/overlib/, Acesso em 18/04/2009.

Bozzon, A., Comai, S., Fraternali, P., and Carughi, G. T. (2006). Conceptual modeling and code generation for rich internet applications. In *ICWE 06: Proceedings of the 6th international conference on Web engineering*, New York, NY, USA. ACM.

Cowin, R. (2009). *Rico*. Disponível em http://openrico.org/, Acesso em 18/04/2009.

da Silva, E. E. and Guimarães, C. C. (2003). Comparação de APIs no Acesso Eficiente a Banco de Dados Heterogêneos na World Wide Web. In *Scientia*. Centro Universitário Vila Velha.

Fuchs, T. (2009). *Script.aculo.us*. Disponível em http://script.aculo.us/, Acesso em 18/04/2009.

Kistener, G. (2004). *Tabstatic*. Disponível em http://phrogz.net/js/Tabtastic/index.html, Acesso em 18/04/2009.

Mochi Media, Inc. (2009). *MochiKit - A lightweight Javascript library*. Disponível em http://mochikit.com/, Acesso em 18/04/2009.

Proietti, V. (2009). *MooTools - a compact javascript framework*. Disponível em http://mootools.net/, Acesso em 18/04/2009.

Resig, J. (2009). *jQuery: The Write Less, Do More, JavaScript Library*. Disponível em http://jquery.com/, Acesso em 18/04/2009.

Ribeiro, D. C., da Silva, E. M. M., Júnior, F. A. S., de Oliveira Rosal, T., and Bogol, M. (2005). Estudo Comparativo das APIs JAX-RPC e JAXM na Construção de Web Services. In *Anais VII Encontro de Estudantes de Informática do Estado do Tocantins, 2005, Palmas*. VII Encontro de Estudantes de Informática do Estado do Tocantins.

The Dojo Foundation (2009). *Dojo Toolkit*. Disponível em http://www.dojotoolkit.org/, Acesso em 18/04/2009.

Turnbull, A. (2007). *Tipster - Twinhelix*. Disponível em http://www.twinhelix.com/dhtml/tipster/, Acesso em 18/04/2009.

Valaer, L. A. and Babb II, R. G. (1997). Choosing a user interface development tool. *IEEE Softw.*, 14(4):29–39.

Yahoo! Inc. (2009). *The Yahoo! User Interface Library*. Disponível em http://developer.yahoo.com/yui/, Acesso em 18/04/2009.

Zorn, W. (2008). *Walterzorn*. Disponível em http://www.walterzorn.com/, Acesso em 18/04/2009.