Enterprise Web 2.0 volgens QAFE

Sneller functionele schermen opleveren

Qafe is een makkelijk te leren framework voor enterprise webapplicaties die op een service-oriented manier werken met bestaande of nieuwe backend-architecturen. Compleet met een extension voor het goeddeels geautomatiseerd migreren van Oracle Forms. Het service-oriented karakter van QAFE, de roadmap en de nieuwe extension, namelijk QAML UI Builder, verdienen een nadere toelichting.

SOA is het antwoord op de eilanden van technologie die in de afgelopen decennia wereldwijd zijn ontstaan. Meer en meer praten verschillende technologieën met elkaar via specifieke protocollen of webservices. Parallel daaraan wordt het belang van een duidelijke scheiding van presentatielaag en de onderliggende code duidelijker. Veel (browser) plug-in technologieën werken al met dit idee, zoals Adobe Flex (MXML) en Microsoft SilverLight (XAML), maar er zijn ook niet-plug-in technologieën actief op dit vlak zoals Google's Web Toolkit met UI Binder en het ZK Framework.

Talen

Al deze technologieën, en vooral de declaratieve presentatiedelen, zijn vrij snel te leren. Maar er is altijd een bijbehorende taal die je onder de knie moet hebben om interactie van componenten en backend integratie te realiseren. In Adobe Flex dien je Action Script te beheersen en eventueel voor de backend Java of een andere taal. Ook bij Google's Web Toolkit wordt veel Java-kennis verwacht.
Ondanks dat deze technologieën nu het moderne webapplicatielandschap definiëren, zijn ze, afhankelijk van de voorkennis, dus nog steeds hoogdrempelig. Oracle-developers die begrijpelijkerwijs niet naar de Java-brugklas willen, blijven aangewezen op Oracle-gereedschap om de brug naar het web te slaan. Maar hoe je het ook wendt of keert, het is en blijft kunst- en vliegwerk aan de voorzijde en een patstelling in de backend.

De reden daarvoor is dat de meeste SOA's er wel in slagen om de eindjes aan elkaar te knopen, maar de services per definitie niet intelligent zijn. De data en de logica blijven in de vertrouwde (in vele gevallen verouderde) omgeving. Tenzij je gaat werken met data-oriented webapplications, zoals in Flex en GWT. Alleen welke van die twee (als we de keuze daartoe zouden beperken) kies je dan?

Keuze

QAFE koos ervoor om niet te hoeven kiezen. Door de XML-programmeertaal QAML te ontwikkelen creëerde Qualogy binnen QAFE een data-oriented business layer met enkele aansprekende voordelen. De techniek is namelijk vanuit de QAML-basis te benaderen voor GWT, Adobe Flex, Adobe AIR (en iedere willekeurige andere Web 2.0-standaard: denk ook aan HTML5-features zoals WebSockets!). De business layer is daarmee daadwerkelijk technologie-onafhankelijk geworden. Bovendien behouden ontwikkelaars de vrijheid om binnen QAFE met services te werken waar gewenst. Ontwikkelaars kunnen doorontwikkelen in hun favoriete technologie. In onvervalst IT-jargon laat dat zich lezen als compleet 'investeringsbehoud'. Je hóeft dus niet alles opnieuw te bouwen.

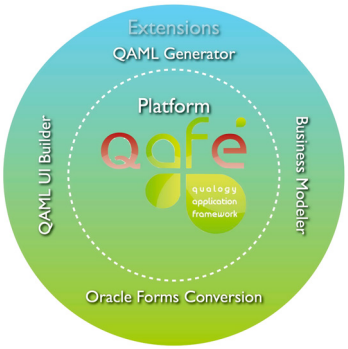
Maar – en dat is waarin QAFE zich onderscheidt – het kán wel.

Voor Oracle- en andere programmeurs komt QAFE zo puntsgewijs hierop neer:

- Toegankelijk maken van moderne presentatietechnologieën zoals GWT en Flex.
- Tegelijkertijd meerdere presentatietechnologieën beschikbaar van één applicatie (handig ook voor locaties met een trage internetverbinding; Adobe Flex/AIR is daar dan beter geschikt dan GWT, vanwege de eenmalige download).
- Eén taal voor integratie met een willekeurige backend, namelijk QAML, QAFE's Extensible Markup Language.
- Application Centric model: alles draait om de applicatie en niet om de technische nitty-gritty.
- RAD (Rapid Application Development) onder meer dankzij XML-code-completion en drag 'n drop QAML UI Builder.

- Service Oriented gedachtengoed zit vanaf de eerste dag verweven in het framework. Integratie met verschillende bronnen binnen één applicatie is dan ook geen punt.
- Open API, voor eigen uitbreidingen.
- Open Source (Apache 2.0) in etappes vanaf 2011 Q1!
- Tooling beschikbaar om productiviteit te verhogen.
- Volledig ondersteund door best practices, design Patterns en vooral ervaring uit de 'real world' (QAFE is een Qualogy-product, wat van oudsher een echte Oracle-club is).

QAFE's ecosystem ziet er op dit moment als volgt uit:



Figuur 1. Het ecosysteem van Qafe.

Het platform is naar smaak uit te breiden met diverse extensions. Aanvullend gereedschap wat nu al verkrijgbaar is, en waar in de toekomst nog de nodige tools bij zullen komen. Het QAFE-team ontwikkelde in de afgelopen maanden een drag 'n drop QAML UI Builder en trof de nodige voorbereidingen voor een langgekoesterde wens: het platform gaat open source!

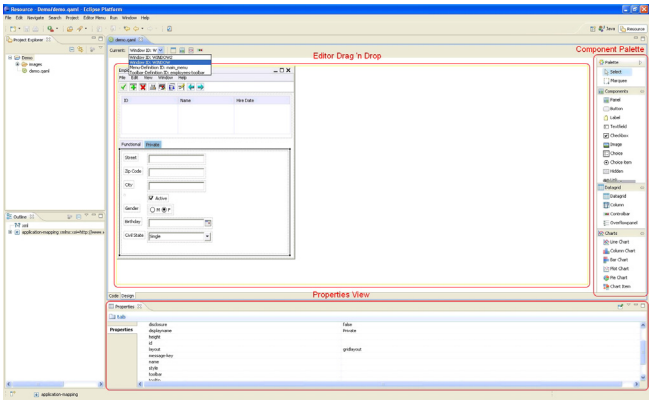
Open source?!

De wens om een open source platform te worden, speelt al een tijdje. Maar hoe richt je dat goed in? Zowel juridisch, bedrijfskundig, als technisch? Na een uitgebreide inventarisatie van de mogelijkheden, is de kogel nu door de kerk. Licentievoorwaarden zijn opgesteld. Het pad naar de eerste community-edition van QAFE is getekend. En begin 2011 is het dan zover. Eerder al: in september 2010, kwam een volledige trial-versie beschikbaar voor iedereen die serieus met QAFE aan de slag wilde.

Qafe is er in drie edities: de Community Edition, de Standard Edition en de Enterprise Edition. Het verschil tussen de Community Edition en de Standard Edition is dat de Community Edition vrij voor iedereen beschikbaar komt onder Open Source licentie van Apache 2.0. De Standard en de Enterprise Editions zijn de versies waarin de onderliggende code is gewaarborgd en waar support op gegeven wordt. De Enterprise Edition heeft wat specifieke zaken die voor Enterprises nodig zijn, zoals support voor Oracle's Virtual Private Database (VPD).

QAML UI Builder

Er is een nieuwe drag 'n drop QAML UI Builder extension beschikbaar. De extension is een eclipse (3.5) plugin die de mogelijkheid biedt om bij wijze van spreken zonder één letter Java, HTML of QAML een webapplicatie te maken. Zolang je het kan 'tekenen' met deze extensie doet QAFE vervolgens de rest en komt de app direct beschikbaar in Flex of GWT (of welke andere ondersteunde Web 2.0-omgeving dan ook). De focus blijft zodoende waar die moet zijn: namelijk bij de applicatie en functionaliteit daarvan.



Figuur 2. Overview van de QAML Builder-plugin.

Figuur 2 laat een overview zien van de QAML UI Builder-plugin voor Eclipse. Dit is de default setup (in Eclipse is het ook mogelijk een eigen setup te creëren). We onderscheiden de volgende onderdelen:

1. De Component Palette (rechts). Hier zijn alle grafische componenten te vinden die QAFE ondersteunt om een presentatielaag te bouwen. De componenten zijn gegroepeerd, alle charting components staan bijvoorbeeld bij elkaar.
2. In het midden zien we de Drag 'n Drop Editor. Hier gebeurt het. Door een component uit de Palette te slepen, is het creëren van een scherm kinderspel.
3. Zodra een component in de Editor geplaatst is, dan bestaat in de Properties View de mogelijkheid om de eigenschappen van deze specifieke component aan te passen. De veranderingen die effect hebben op de look-and-feel zijn direct zichtbaar in de Editor (zoals disabled, if true, dan wordt het grayed-out).
4. Links in Eclipse is tot besluit de Project Explorer waar alle bij het project behorende QAML-files te vinden zijn.

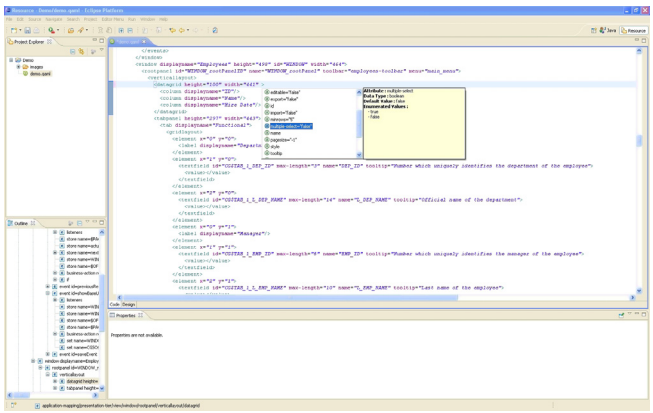
In figuur 2 is te zien dat er al een Window gecreëerd is. Eentje met een menu en daaronder een toolbar met een aantal items, deze zijn respectievelijk gedefinieerd als Menu-Definition en Toolbar-Definition (allen herbruikbare componenten). Een Window kent default een RootPanel. Op Panels kunnen weer andere componenten zoals aanvullende Panels geplaatst worden. In dat geval geef je aan in welke oriëntatie (layout) dat moet gebeuren:

- absolute layout
- auto layout
- border layout
- grid layout
- horizontal layout
- vertical layout

In het voorbeeld is een Datagrid en een TabPanel-component in de Window gesleept. Het Tab-component is niets anders dan een Panel, welke meerdere componenten kan hebben en dus ook een layout heeft om zijn componenten, oftewel zijn kinderen, te positioneren. In dit geval kent de TabPanel twee Tab-componenten op de RootPanel. Het Tab-component met de titel 'Private' heeft een gridlayout met 2 kolommen en 7 rijen en bevat Label, TextField (property type is text), Checkbox, Choice met 2 Choice Items (M en F), TextField (property type is date) en DropDown met DropDown Items.

Code editor

Tot zover is het scherm volledig gebouwd in de drag 'n drop QAML UI Builder. Mocht een developer alsnog in het totaal aan gegenereerde code willen kijken en die aanpassen, dan is het mogelijk om in de Editor View het Code-tabblad (linksonder in de Editor view) te selecteren. Om vervolgens de aanpassingen direct te maken in QAML. Code changes die handmatig zijn gemaakt, zijn direct zichtbaar in de Design-tab.



Figuur 3. De textmode van de QAML UI Builder.

Handmatig code bewerken in QAML is makkelijker gemaakt. Het QAFE-team heef de structuur en met name de overzichtelijkheid (en herbruikbaarheid van Panel-Definitions e.d.) aanzienlijk verbeterd. De drag 'n drop UI Builder biedt uitkomst bij twijfel. En werkend in de Code-tab geldt auto-completion a.h.v. de XSD, zodat editen van een file flink wat sneller gaat.

Andere features

Images met een URL van internet zijn direct zichtbaar (en resizebaar). De entries van een dropdown zijn ook via drag 'n drop te plaatsen, te herschikken, etc. Layouts zijn met één klik gewijzigd. En zo zijn er in de QAML UI Builder nog meer features te vinden die voor een hogere productiviteit zorgen. Grotere applicaties zullen bijvoorbeeld meer Windows hebben of meer herbruikbare Panel-componenten (denk aan wizards). Eenmaal aangemaakt zijn die voortaan standaard via de dropdown linksboven in de Editor te vinden. Eigen Menu-Definitions, Panel-Definitions en Toolbar-Definitions worden zo ideale bouwstenen om nieuwe schermen mee op te tuigen en/of om alle schermen met een bepaalde component in één moeite mee aan te passen.

De voordelen

Het mooie van drag 'n drop editing is dat er geen haakjes o.i.d. worden gemist wat bij het handmatig editen van XML nog wel eens wil gebeuren. Ook om het overzicht te houden is drag 'n drop-editing prettig, zeker bij complexere, grotere applicaties. Bovendien wordt veel van wat technisch wel en niet mag via de UI Editor afgevangen. Het gemak van deze tool betekent zodoende dat ontwikkelaars sneller functionele schermen opleveren, waardoor er meer tijd over is voor de functionaliteit áchter de schermen. En dat werkt wel zo prettig. Zeker in de wetenschap dat je dankzij de technologie-onafhankelijke structuur van QAFE, bij de komst van nieuwe webtechnologieën niet weer opnieuw hoeft te beginnen.

Referenties

Voor meer informatie zie:
http://www.qafe.com en http://www.qualogy.com

Rokesh Jankie (boven) en Reyco Kha zijn senior consultants bij Qualogy Consultancy.