Dynamisches Arbeiten mit Grafiken innerhalb von APEX

Tobias Arnhold
Tobias Arnhold – IT Consulting
Heppenheim

Schlüsselworte
APEX, RaphaelJS, Visualisierung, Javascript, HTML5, SVG, APEX-AT-WORK

Einleitung
In dem Vortrag zum Thema „Dynamisches Arbeiten mit Grafiken innerhalb von APEX“ wird eine Möglichkeit gezeigt Grafiken dynamisch zur Laufzeit, abhängig von den Bewegungsdaten, zu generieren. Grundidee dieser Technik ist es, auf ein vorhandenes Hintergrundbild SVG-Objekte zu projizieren und somit dem Anwender ein visuelles Gadget zur Verfügung stellen, anhand dessen er zum Beispiel schnell Veränderungen in den Daten auffassen kann. Als Beispiel dient in diesem Vortrag eine Anzeige zur Hafenbelegung. Das Hintergrundbild stellt hierbei eine schematische Abbildung des Hafens dar. Die Boote sind die dynamisch generierten und positionierten SVG-Objekte, welche an den jeweiligen Anlegestellen angezeigt werden sollen.

Dynamisches Arbeiten mit Bilder in APEX
Die Technik im Vortrag bezieht sich auf die Javascript Bibliothek RaphaelJS (Raphaël).
RaphaelJS ist eine Vectorgrafik-Library, die unterschiedliche Formen im Browser malen kann und dynamische Interaktionen mit den gezeichneten Elementen zulässt.

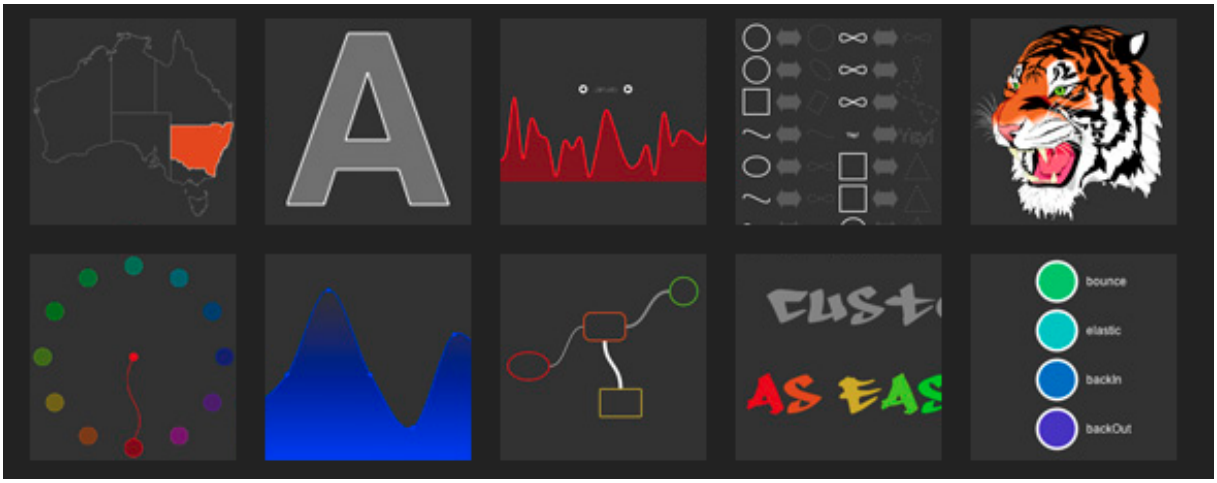


Abb. 1: RaphaelJS Beispiele

Die Einbindung dieser Technik entpuppt sich als sehr einfach. Da nur sehr wenig CSS und HTML Code benötigt wird und die komplette Codergenerierung über Javascript gesteuert wird. Der Javascript Teil wiederholt sich je generiertes SVG-Objekt, wodurch eine automatische Generierung des JS-Codes sehr einfach ist.

In den folgenden Code-Segmenten wird auf das Eingangs erwähnte Hafenbeispiel referenziert.

Schritt 1: RaphaelJS Implementierung im APEX Page Header

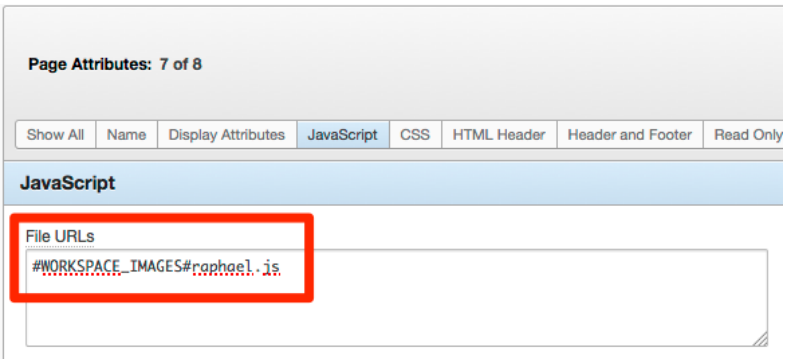


Abb. 3: RaphaelJS Integration in APEX

Schritt 2: CSS Styles hinterlegen

```
<style>
#wrapper {
position: relative;
/* Definition der genauen Bild Groesse und Breite */
width: 397px;
height: 398px;
padding: 0;
outline: 1px solid #999;
}
#wrapper img {
position: absolute; top: 0; left: 0;
}
#canvas{
position: absolute;  top: 0; left: 0;
}
</style>
```

Schritt 3: HTML Elemente erstellen

```
<div id="wrapper">
  <img src="#WORKSPACE_IMAGES#Hafen.png">
  <div id="canvas"></div>
</div>
```

Schritt 4: Javascript Code für die Generierung von zwei SVG-Objekten

```
<script>
$( document ).ready(function() {
  var canvas = Raphael(document.getElementById("canvas"), 397, 398);
  var img1 = canvas.image("#WORKSPACE_IMAGES#yacht.png",
                          180, 90, 26, 19
                         )
                         .glow({width:1,opacity:0.8});
  var img2 = canvas.image("#WORKSPACE_IMAGES#cargo_ship.png",
                          100, 276, 40, 19
                         )
                         .glow({width:1,opacity:0.8});
});
</script>
```

Die beschriebene Technik generiert kein komplettes SVG-Objekt, sondern erstellt zwei kleine SVG-Objekte (Boote) und legt diese auf das bestehende PNG-Bild. Dadurch vermindert sich der Initiale Konfigurationsaufwand immens und die Wiederverwendbarkeit des Codes erhöht sich ebenfalls.

Info: In einem kompletten SVG-Objekt wäre auch der Hintergrund (Hafen) dynamisch erstellt wurden.

Das Ergebnis des gezeigten Codes sieht dann so aus:

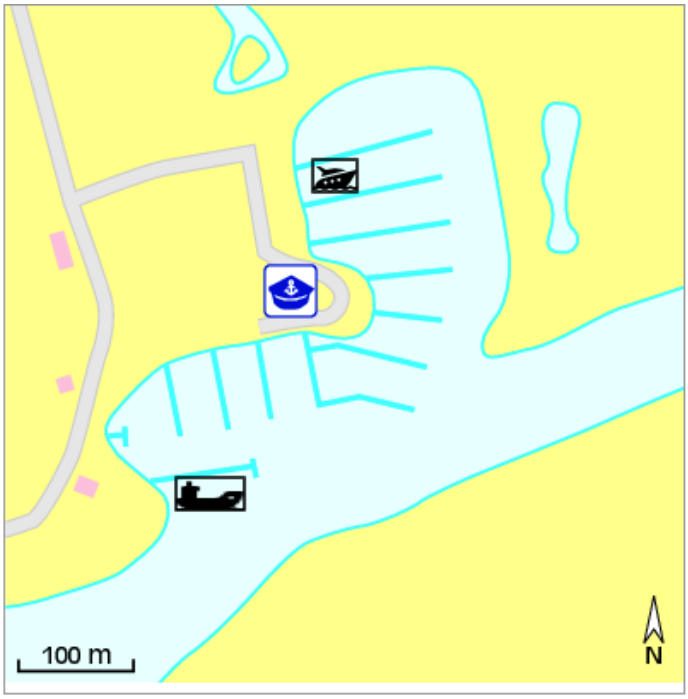
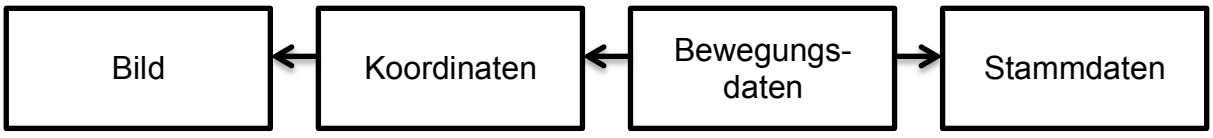


Abb. 3: Hafen Beispiel

Die Basis für die gezeigte Technik, bildet immer ein Bild und wird anschließend durch Koordinaten- und Bewegungsdaten vervollständigt.



Die Komplexität der Detaildarstellung kann beliebig erhöht werden. Details dazu werden mit unterschiedlichen anderen Beispielen im Vortrag näher erläutert. Außerdem wird im Vortrag auf die Interaktion zwischen APEX und RaphaelJS näher eingegangen.

Zusammenfassend ergeben sich folgende Vor- und Nachteile.

Vorteile:

- Visuelle Darstellung von spezifischen Business Prozessen
- Code kann beliebig komplex skaliert werden
- Geringer Entwicklungsaufwand im Vergleich zum Nutzen innerhalb der Anwendung
- Wiederverwendbarkeit der Grundlogik, der Tabellenstrukturen und der Prozesse

- Hohe Browser Kompatibilität (IE 6+, Firefox 3+, Safari 3+, Chrome 5+)
- HTML only (Kein Flash, Java oder Silverlight)
- Open Source (MIT Lizenz)
- Einfache Implementierung der RaphaelJS Engine
- Hervorhebung / Kennzeichnung von Warnschwellen innerhalb der generierten Grafik

Nachteile:

- Zusätzliche Komponente innerhalb der APEX Anwendung
- Keine beliebige Skalierung der Darstellung möglich, da die Grundlage ein nicht skalierbares Bild ist.
  Info: Die Aussage bezieht sich auf das bereitgestellte Beispiel.
- Viel wiederholender Javascript Code, wenn sehr viele SVG-Objekte erstellt werden. Dies kann in älteren Browsern zu Performance Problemen führen (IE 6-7).

Verwendung von ThirdParty Komponenten:

Verwendete Bilder stammen von http://icons8.com/ und http://www.marinafuehrer.adac.de/

Weiterführende Infos zu RaphaelJS finden Sie unter: http://raphaeljs.com/

Kontaktadresse:
Tobias Arnhold
Tobias Arnhold – IT Consulting
Im Mantel 15
D-64646 Heppenheim

Telefon: +49 (0) 157-8484 3005
E-Mail tobias-arnhold@hotmail.de
Internet: www.apex-at-work.de