Grafiken dynamisch erzeugen in HTML5

Canvas

kurz & gut

David Flanagan
Übersetzung von Lars Schulten

O'REILLY®

Inhalt

Canvas-Referenz

Dieser Teil des Buchs ist eine Referenz, die das <canvas>-Tag und die damit verbundenen Klassen dokumentiert. Die Referenz ist alphabetisch sortiert, und Methoden werden über ihren vollständigen Namen eingeordnet, der die Namen der Klassen einschließt, die sie definieren. Wenn Sie mehr über die getContext()-Methode erfahren wollen, müssen Sie beispielsweise Canvas.getContext() nachschlagen. Wollen Sie Näheres über die arc()-Methode wissen, schlagen Sie unter CanvasRenderingContext2D.arc() nach. Weil der Name dieser Klasse so lang ist, wird er in dieser Referenz mit »CRC« abgekürzt.

Der größte Teil dieses Kapitels dokumentiert Methoden des Canvas Rendering Context2Ds (unter dem Namen CRC), aber es werden auch Canvas, CanvasGradient, CanvasPattern, ImageData und TextMetrics behandelt.

Canvas — ein HTML-Element für skriptbasiertes Zeichnen

Eigenschaften

String width, height
: Diese Eigenschaften spiegeln die width- und height-Attribute des <canvas>-Tags und geben die Maße des Canvas-Koordinatenraums an. Die Standardwerte für width und height sind 300 bzw. 150.

  Wenn die Größe des Canvas-Elements nicht anderweitig in einem Stylesheet oder mit einem style-Attribut angegeben wird, bestimmen diese width- und height-Eigenschaften auch

den Platz, den das Canvas-Element auf dem Bildschirm einnimmt.

Wird eine dieser Eigenschaften gesetzt (sogar auch wieder auf den gleichen Wert), wird das Canvas geleert, und alle seine Grafikattribute werden auf die Standardwerte zurückgesetzt.

Methoden

`getContext()`
: Gibt ein Kontext-Objekt zurück, über das Sie auf das Canvas zeichnen können. Übergeben Sie den String `2d`, erhalten Sie ein CanvasRenderingContext2D-Objekt für zweidimensionale Zeichnungen.

  Übergeben Sie den String `webgl`, erhalten Sie ein WebGLRenderingContext-Objekt für 3-D-Darstellungen in Browsern, die das unterstützen. WebGLRenderingContext ist noch nicht standardisiert und wird in diesem Buch nicht dokumentiert.

`toDataURL()`
: Gibt eine `data:`-URL zurück, die das Bild auf dem Canvas darstellt.

Beschreibung

Das Canvas-Objekt repräsentiert ein HTML-Canvas-Element. Es hat kein eigenes Verhalten, definiert aber eine API, die skriptbasierte clientseitige Zeichenoperationen ermöglicht. Sie können `width` und `height` direkt auf diesem Objekt angeben und mit `toDataURL()` ein Bild aus dem Canvas abrufen, aber die eigentliche Zeichen-API wird durch ein separates »Kontext-Objekt« gestellt, das von der Methode `getContext()` zurückgeliefert wird. Siehe `CRC`.

Das `<canvas>`-Tag wurde in Safari 1.3 eingeführt und wird mit HTML5 standardisiert. Es wird von allen aktuellen Versionen von Firefox, Safari, Chrome und Opera unterstützt. Es wird ebenfalls vom Internet Explorer 9 unterstützt und kann in früheren IE-Versionen mit der Open Source-Bibliothek ExplorerCanvas unter *http://code.google.com/p/explorercanvas/* emuliert werden.

Siehe auch

`CRC`

Canvas.getContext() liefert einen Kontext zum Zeichnen auf das Canvas

Überblick

```
Object getContext(String contextID)
```

Argumente

contextID
: Dieses Argument gibt die Art der Zeichnung an, die auf dem Canvas erstellt werden soll. Übergeben Sie `2d`, um ein CanvasRenderingContext2D-Objekt zu erhalten, mit dem Sie zweidimensionale Zeichnungen ausführen können.

Rückgabewert

Ein Objekt, mit dem Sie in das Canvas-Element zeichnen können. Wenn Sie den String `2d` übergeben, ist das ein CanvasRenderingContext2D-Objekt für 2-D-Zeichnungen.

Beschreibung

Es gibt nur ein CanvasRenderingContext2D-Objekt pro Canvas-Element. Mehrfache Aufrufe von `getContext("2d")` liefern also dasselbe Objekt.

HTML5 standardisiert das `2d`-Argument für diese Methode und definiert keine weiteren gültigen Argumente. Ein anderer Standard, WebGL, für 3-D-Grafiken befindet sich in der Entwicklung. In unterstützenden Browsern können Sie dieser Methode den String `webgl` übergeben, um ein Objekt zu erhalten, das 3-D-Darstellung ermöglicht. Beachten Sie allerdings, dass das CanvasRenderingContext2D-Objekt der einzige Zeichenkontext ist, der in diesem Buch dokumentiert wird.

Siehe auch

`CRC`

Canvas.toDataURL() liefert ein Canvas-Bild als data:-URL

Überblick

```
String toDataURL()
String toDataURL(String type, Parameter...)
```

Argumente

type
: Ein String, der den MIME-Typ für das zu nutzende Bildformat angibt. Wenn dieses Argument weggelassen wird, wird der Standardwert `image/png` verwendet. Das PNG-Format ist das einzige Format, das konforme Implementierungen unterstützen müssen.

Parameter...
: Bei anderen Bildtypen als PNG können zusätzliche Argumente angegeben werden, die Kodierungsinformationen enthalten. Ist *type* `image/jpeg`, sollte das zweite Argument beispielsweise eine Zahl zwischen 0 und 1 sein, die die Qualitätsstufe des Bilds angibt. Aktuell sind keine weiteren Parameter standardisiert.

Rückgabewert

Ein String, der eine PNG-Darstellung der Canvas-Bitmap als `data:`-URL kodiert enthält.

Beschreibung

`toDataURL()` liefert den Inhalt der Canvas-Bitmap in einer URL-Form, die leicht in einem `<img>`-Tag genutzt oder über ein Netzwerk übertragen werden kann.

Zur Vermeidung von Cross-Origin-Sicherheitslöchern funktioniert `toDataURL()` nicht mit `<canvas>`-Tags, die nicht »herkunftsrein« sind. Ein Canvas ist nicht herkunftsrein, wenn in es (direkt mit `drawImage()` oder indirekt über ein CanvasPattern) ein Bild gezeichnet wurde, das von einer anderen Quelle stammt als das Dokument, das das Canvas enthält.

Beispiel

```
// Den Inhalt des Canvas in ein img-Element kopieren
// und dieses Bild an das Dokument anhängen.
var canvas = document.getElementById("my_canvas");
var image = document.createElement("img");
image.src = canvas.toDataURL();
document.body.appendChild(image);
```

Siehe auch

`CRC.getImageData()`

CanvasGradient — ein Farbverlauf zur Verwendung in einem Canvas

Methoden

`addColorStop()`
: Gibt eine Farbe und eine Position für den Verlauf an.

Beschreibung

Ein CanvasGradient-Objekt repräsentiert einen Farbverlauf, der den `strokeStyle`- und `fillStyle`-Eigenschaften eines CanvasRenderingContext2D-Objekts zugewiesen werden kann. Die `createLinearGradient()`- und `createRadialGradient()`-Methoden des CanvasRenderingContext2D-Objekts liefern beide CanvasGradient-Objekte.

Haben Sie ein CanvasGradient-Objekt, nutzen Sie `addColorStop()`, um anzugeben, welche Farben an welchen Positionen des Verlaufs erscheinen sollen. Zwischen den von Ihnen angegebenen Positionen werden die Farben interpoliert, um einen stetigen Verlaufs zu erstellen. Wenn Sie keine Farbpositionen angeben, wird der Verlauf durchgängig schwarz.

Siehe auch

`CRC.createLinearGradient()`, `CRC.createRadialGradient()`

CanvasGradient.addColorStop() — gibt eine Farbe auf dem Verlauf an

Überblick

```
void addColorStop(float offset, String color)
```

Argumente

`offset`
: Ein Fließkommawert im Bereich zwischen 0.0 und 1.0, der die Position zwischen Start- und Endpunkt des Verlaufs angibt. Eine Verschiebung von 0 entspricht dem Startpunkt und eine Verschiebung von 1 dem Endpunkt.

`color`
: Gibt die an dieser Position anzuzeigende Farbe als CSS-Farbstring an. Farben an anderen Punkten auf dem Verlauf werden auf Basis dieser Farbe und der anderen Stoppfarben interpoliert.

Index

T

V

W

Z