Guida HTML5: il Canvas - Parte 4

Benvenuto alla quarta e ultima parte di questa guida dedicata al Canvas. Oggi tratteremo alcune tecniche avanzate che ci permetteranno di gestire le ombre, i pattern, i gradient e di modificare immagini.

Modificare immagini

Come prima cosa, settiamo il nostro Canvas associato a uno specifico ID affinché riproduca un'immagine nella nostra pagina:

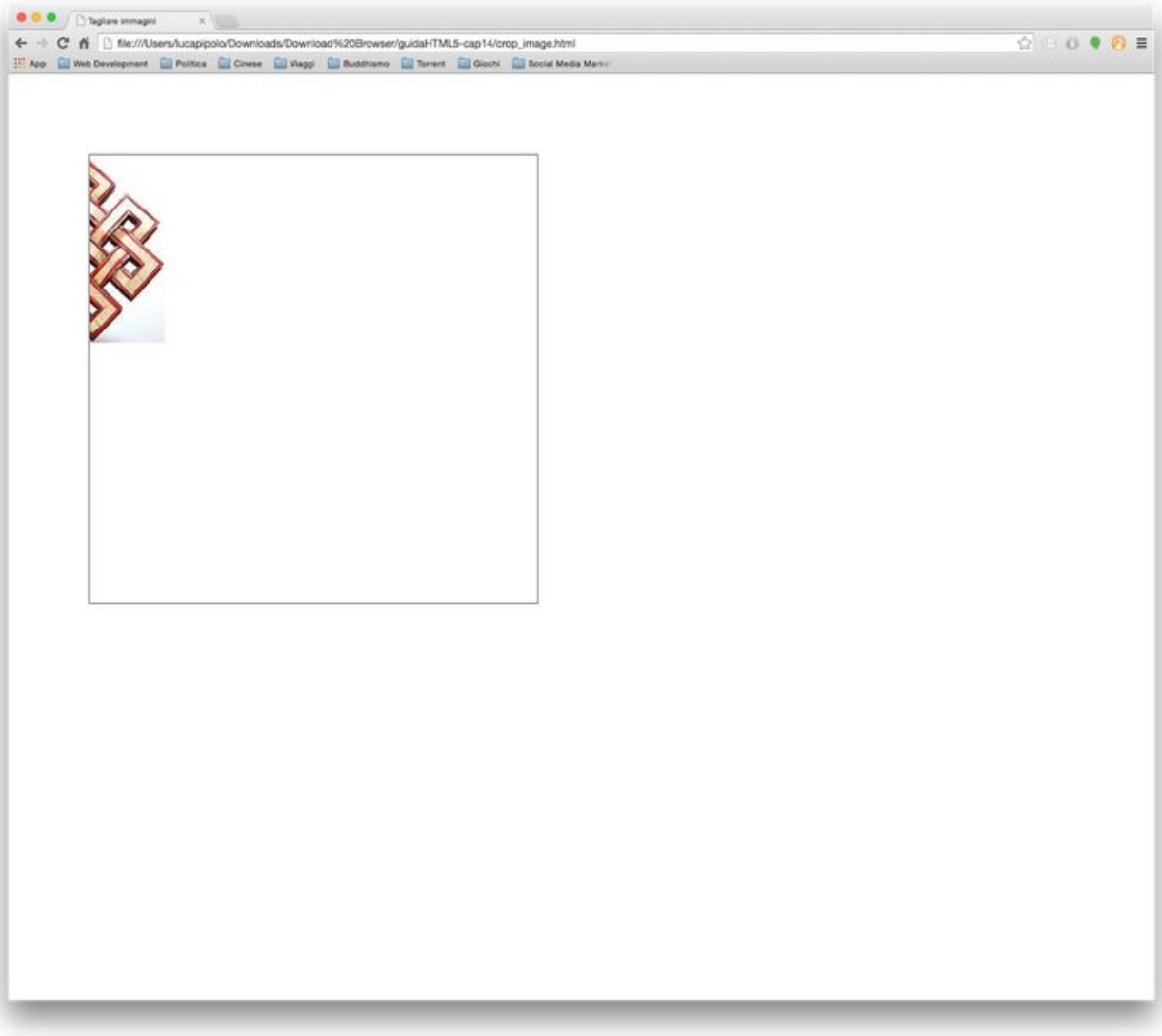
All'interno del codice JS, invece, andiamo a scrivere:

Analizzando il codice appena creato capiamo che, dopo aver associato la variabile *canvas* all'elemento con l'id *canvas_immagine* e dopo aver settato il *context* su *2d*, creiamo un elemento immagine che avvierà una funzione, la quale conterrà il metodo *drawImages*. I parametri passati al metodo *drawImages* ci permettono di modificare rispettivamente le coordinate di partenza, la larghezza e l'altezza. Ad esempio:

corrisponderà a un'immagine che, a partire dalle coordinate *0*, *0*, misurerà 201px di larghezza e 251px di altezza.

Proviamo adesso a tagliare un'immagine all'interno di Canvas, quello che solitamente viene definito *crop*. Per far ciò ci basterà specificare i parametri *sx*, *sy*, *swidth* e *sheight*; essi rappresentano rispettivamente le coordinate di partenza, la larghezza e l'altezza dell'immagine visualizzata a dispetto dell'originale. Quindi scrivendo:

avremo, come risultato:



Ricapitolando, la sintassi da utilizzare per "croppare" un’immagine è:

Ombre

In Canvas è possibile, tra le altre cose, gestire le ombre delle forme. Iniziamo col disegnare un rettangolo con il metodo che già conosciamo.

A questa forma associamo adesso un’ombra. Per farlo scriviamo:

Analizziamo quanto appena scritto:

- *shadowColor* si occupa di settare un colore per l'ombra. Può contenere qualsiasi tipo di colore espresso in codice esadecimale, oppure può ottenere anche colorazioni con trasparenza attraverso l'rgba.
- *shadowsBlur* gestisce quella che, in italiano, è denominata sfocatura. Più basso è il valore assegnato a questo parametro, maggiormente risulterà definita l'ombra; contrariamente maggiore è il valore, maggiore la sfocatura. Generalmente è sconsigliabile utilizzare valori minori di 3.
- *shadowOffsetX* e *shadowOffsetY* determinano la grandezza dell'ombra in pixel, rispettivamente a destra e in basso. Per impostare l'ombra in direzione opposta, quindi verso sinistra e in alto, ti basterà impostare valori negativi.

Pattern & Gradient

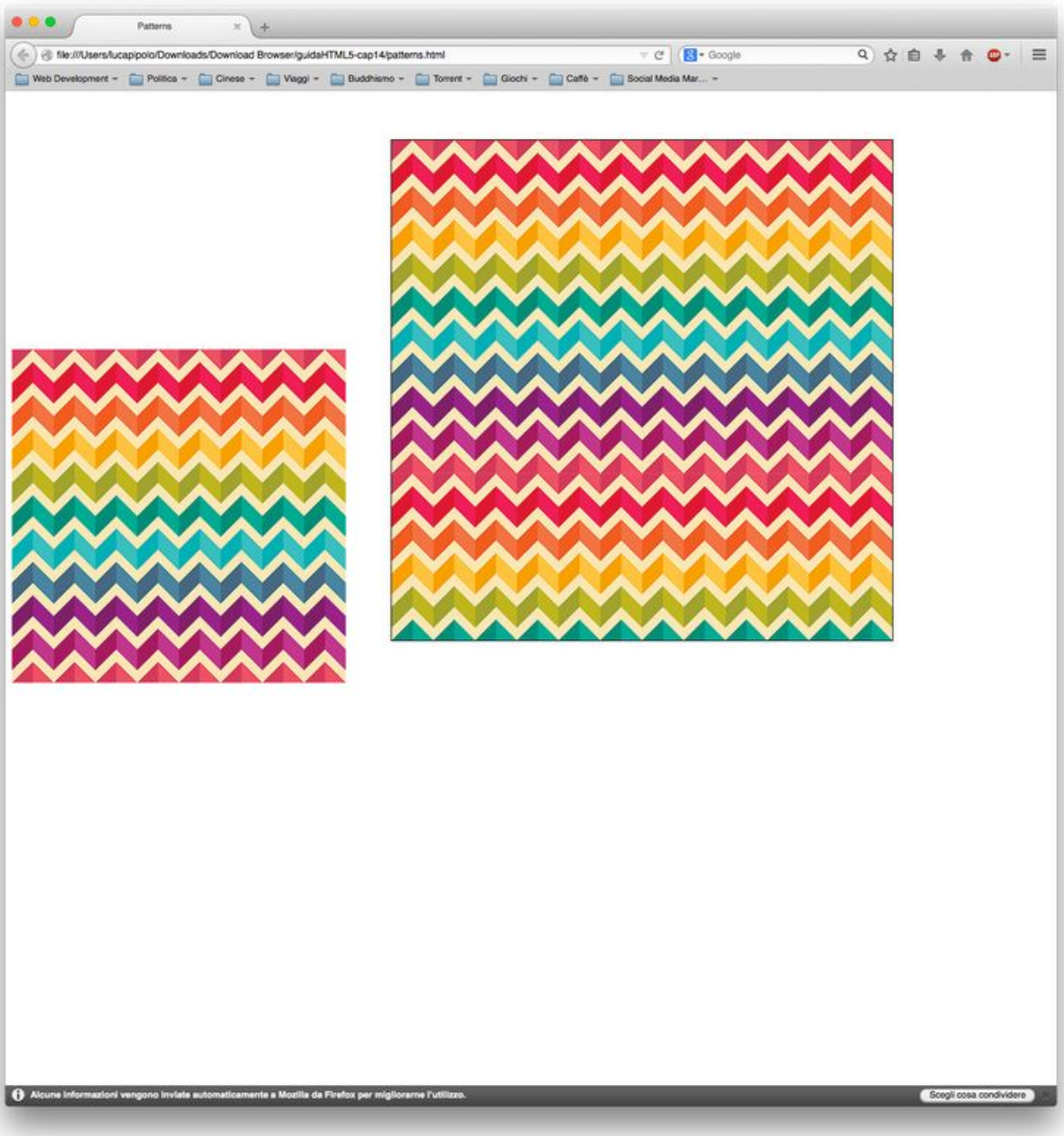
Nelle scorse lezioni abbiamo imparato come riempire le forme create in Canvas con colori solidi o semitrasparenti, ma vi sono ben altre possibilità di vivacizzare i nostri elementi!

Il primo di questi metodi è rappresentato dai pattern. Iniziamo scegliendo un'immagine che ben si possa prestare alla creazione di un pattern e aggiungiamola attraverso il consueto tag *img* alla nostra pagina.

Proseguiamo creando le due variabili, *img* e *pattern*, associate all'immagine e al pattern che stiamo creando da essa.

Infine, completiamo il processo attribuendo al metodo *fillStyle* la variabile pattern appena creata invece di un solito colore solido.

Il risultato finale dovrebbe apparirti pressappoco così:



L’immagine a sinistra rappresenta l’originale, mentre quella a destra è il risultato del pattern in Canvas. Ho sperimentato personalmente che l’ultima versione di Google Chrome non supporta questo metodo quindi, prima di utilizzarlo, informatevi su quali browser siano in grado di visualizzarlo correttamente.

Lo stesso concetto che abbiamo utilizzato per i pattern, può essere associato ai gradient. Per far ciò utilizziamo il metodo *createLinearGradient*, passando ad esso quattro parametri indicanti le coordinate *x* e *y* di partenza e quelli di fine.

Settiamo i colori, in questo caso tre, del gradient.

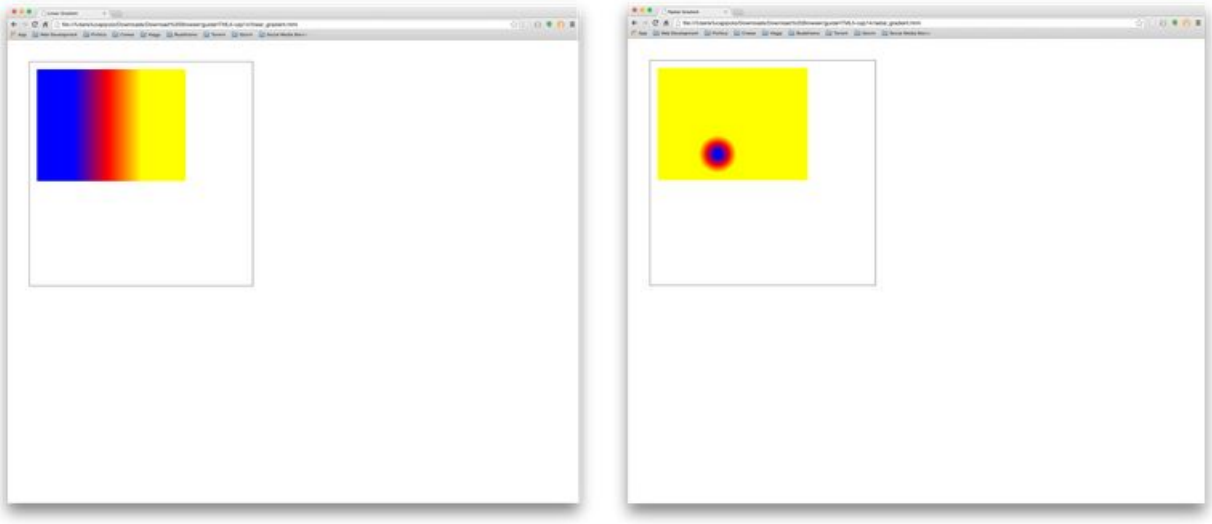
E utilizziamolo come *fillStyle* di un rettangolo.

I numeri passati come parametro a *addColorStop* rappresentano l'offset dei rispettivi colori all'interno di un range che va da 0 a 1.

Esiste anche un metodo molto simile che ci permette di generare gradient radiali, anziché creare gradient lineari.

L'unica differenza di *createRadialGradient* è quella di accettare sei parametri, invece di quattro, i quali sono rispettivamente: coordinate di partenza *x* e *y*, raggio di partenza, coordinate di arrivo *x* e *y*, e raggio d'arrivo.

Qui di seguito puoi osservare come si presenta il risultato di entrambi i metodi.



Conclusione

Nel corso di questa lezione abbiamo imparato a utilizzare gli ultimi metodi riguardanti il Canvas. Dal prossimo articolo, invece, inizieremo ad affrontare l'ultimo discorso di questa ormai lunga guida, ovvero gli strumenti utili alla creazione di applicazioni web: il data storage, il supporto offline e le comunicazioni con il web server.

Come al solito, per rivedere il codice completo di questa lezione, fai riferimento all'archivio .zip che trovi qui.

GUIDA HTML5: GLI ARTICOLI

1) Guida HTML5:Introduzione
2) Guida HTML5: la prima pagina
3) Guida HTML5: la struttura
4) Guida HTML5: Immagini e outlines
5) Guida HTML5: nuovi elementi semantici
6) Guida HTML5: i form – Parte 1
7) Guida HTML5: i form – Parte 2
8) Guida HTML5: i form – Parte 3
9) Guida HTML5: i form – Parte 4
10) Guida HTML5:i tag audio e video – parte 1
11) Guida HTML5:i tag audio e video – parte 2
12) Guida HTML5: I player video
13) HTML5: Il Canvas – Parte 1
14) Guida HTML5: Il Canvas - Parte 2
15) HTML5: Il Canvas - Parte 3
16) HTML5: Il Canvas - Parte 4
17) HTML5: Web storage