Capitolo 1

Introduzione allo sviluppo iOS con tecnologie web

L'introduzione di iPhone e la presentazione successiva di iPod touch e iPad hanno rivoluzionato il modo con il quale le persone interagiscono con i dispositivi portatili. Gli utenti non devono più utilizzare una tastiera per muoversi tra le schermate o visualizzare pagine web "semplificate". Questi device mobili hanno introdotto modalità di input tramite touchscreen, un design di interfaccia rivoluzionario e un browser web ricco di funzioni, il tutto direttamente nel palmo di una mano.

Le potenzialità della piattaforma hanno scatenato la creatività della comunità di sviluppatori. Le applicazioni native ricevono probabilmente la maggior parte delle attenzioni, ma si possono comunque realizzare app per dispositivi iOS anche senza scrivere una sola riga in Objective-C. In realtà, il browser Safari su iOS fornisce una piattaforma di sviluppo delle applicazioni web irresistibile, da impiegare per realizzare app personalizzate per il sistema iOS sfruttando tecnologie web già conosciute.

In questo capitolo

- **Alla scoperta di Safari sulla piattaforma iOS**
- **Funzionalità principali di Safari per lo sviluppo web**
- **Quattro soluzioni per lo sviluppo di web app per iOS**
- **Il dito non è un mouse**
- **Limitazioni e vincoli**
- **Impostare l'ambiente di sviluppo in una rete locale**

Alla scoperta di Safari sulla piattaforma iOS

Ogni applicazione web viene eseguita nel browser Safari presente nel sistema iOS, il cui funzionamento si basa su standard web che includono:

- HTML/XHTML (tipi di documento mobile profile HTML 4.01 e XHTML 1.9, XHTML);
- CSS (CSS 2.1 e parzialmente CSS3);

- JavaScript (ECMAScript 3 – ECMA 262 –, JavaScript 1.4);
- AJAX (per esempio XMLHTTPRequest);
- SVG (*Scalable Vector Graphics*) 1.1;
- Media tag HTML5;
- Tecnologie Ancillary (video e audio media, PDF e altro ancora).

Safari su iOS (cui si farà riferimento in questo libro indicando indifferentemente *Safari* oppure *Safari su iOS*) diventa in questo modo la piattaforma di sviluppo delle applicazioni e la shell nella quale devono funzionare le app (Figura 1.1).

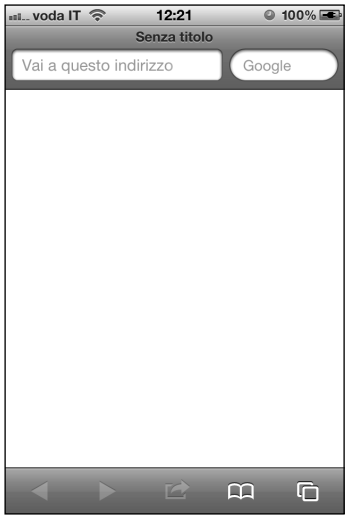

Figura 1.1

Safari è realizzato con lo stesso motore per browser WebKit open source utilizzato in Safari per OS X e per Windows. Nonostante l'intera famiglia di browser Safari si basi su un framework comune, è utile pensare a Safari su iOS come a un parente stretto della controparte per Mac o Windows, piuttosto che un gemello identico a una delle altre versioni. Per esempio, Safari su iOS non garantisce la piena compatibilità con le funzioni CSS o JavaScript che sono invece presenti nella controparte per computer desktop. Va inoltre ricordato che Safari su iOS mette a disposizione un numero limitato di impostazioni configurabili dall'utente. Come si può vedere nella Figura 1.2, gli utenti possono attivare o meno il supporto di JavaScript, dei plug-in e il blocco di pop-up e possono anche scegliere se accettare o meno i cookie indiscriminatamente, accettare i cookie solo dai siti che stanno visitando oppure escludere la presenza di cookie. Questa stessa schermata permette infine di cancellare manualmente la cronologia, i cookie e la cache. È ovvio ritenere che le app native e le web app non siano la stessa cosa, sia dal punto di vista dello sviluppatore sia dell'utente finale. Per lo sviluppatore la differenza fondamentale è costituita dal linguaggio di programmazione, in quanto si devono utilizzare le tecnologie web al posto di Objective-C.

Figura 1.2

Ci sono implicazioni anche dal punto di vista dell'utente finale, tra cui quelle indicate di seguito.

- Prestazioni: un'applicazione web basata su Safari non sarà reattiva come un'applicazione nativa, tenendo conto della necessaria interpretazione dei linguaggi di programmazione da parte del sistema e del fatto che l'applicazione sta sfruttando il collegamento a reti Wi-Fi e 3G. Va ricordato inoltre che alcuni modelli di iPad e tutti gli iPod touch supportano esclusivamente l'accesso a reti Wi-Fi. Nonostante questi limiti tecnologici, si possono comunque ottimizzare le applicazioni in modo da ottenere prestazioni accettabili. La Tabella 1.1 mostra le prestazioni in termini di larghezza di banda delle connessioni Wi-Fi, 3G e delle meno recenti reti EDGE.

Tabella 1.1 Prestazioni delle connessioni in rete.

Rete	Larghezza di banda
Wi-Fi	Fino a 450 Mbps
3G	Fino a 7,2 Mbps
EDGE	70-135 Kbps, picchi fino a 473,6 Kbps

- Modalità di avvio: le applicazioni native devono essere avviate dalla schermata *Home* del device iOS (Figura 1.3). La versione originale di iOS non consentiva di avviare le web app in questo modo, il che implicava la necessità di accedervi esclusivamente tramite l'elenco dei *Preferiti* di Safari. Per fortuna, le versioni più recenti di iOS permettono agli utenti di aggiungere "Web Clip" relative a web app nella schermata *Home*, come nella Figura 1.4, che riporta l'icona di *iDocs*.

Figura 1.3

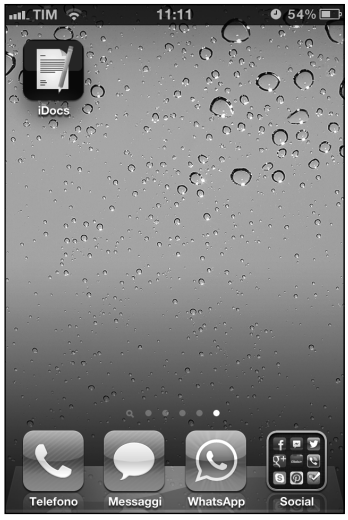

Figura 1.4

- Interfaccia utente o UI (*User Interface*): le applicazioni iOS native sono spesso conformi alle linee guida di Apple per il design relativo alle interfacce utente. Quando si progetta una web app non ci si deve mai sentire obbligati a ricreare una UI il cui aspetto sia simile a un'applicazione nativa; nel contempo, si deve creare una UI adatta idealmente a un device mobile e dotato di touchscreen. L'utilizzo di framework open source e delle tecnologie web standard consente fortunatamente di combinare soluzioni basate su HTML, CSS e JavaScript. Le Figure 1.5 e 1.6 mettono a confronto il design di una UI di un'applicazione nativa e quella di una web app basata su Safari.

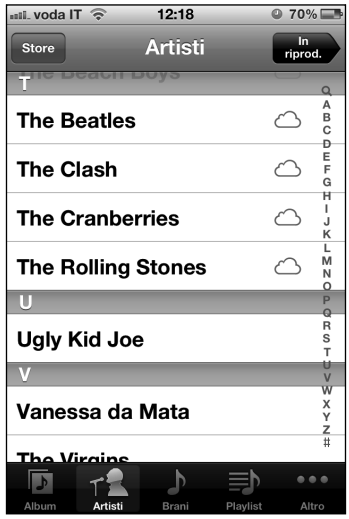

Figura 1.5

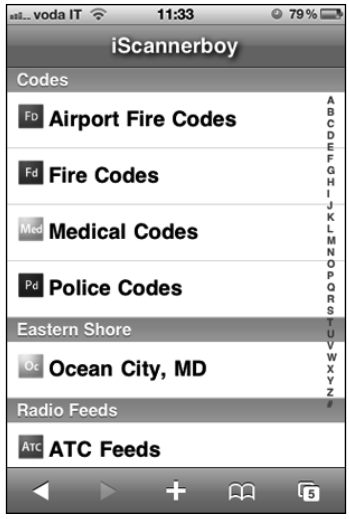

Figura 1.6

Va inoltre ricordato che gli ultimi aggiornamenti di iOS consentono di utilizzare i metatag per nascondere gli elementi UI del browser Safari e ciò permette di emulare fondamentalmente il *look and feel* di un'applicazione nativa, come si può vedere nella Figura 1.7.

Funzionalità principali di Safari per lo sviluppo web

Le versioni più recenti di iOS offrono molte funzionalità utili per chi deve sviluppare web app, come si può vedere nell'elenco che segue.

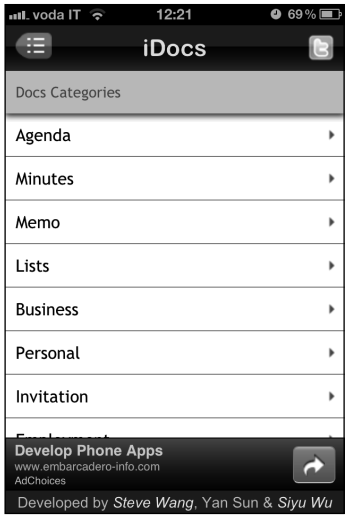

Figura 1.7

- Geolocalizzazione: Safari su iOS supporta le funzioni di geolocalizzazione HTML5, che permettono a JavaScript di interagire con il servizio GPS di iPhone o iPad per conoscere la posizione attuale del device, come mostrato nelle Figure 1.8 e 1.9. È quindi possibile realizzare applicazioni in grado di trasmettere la posizione di un device iOS con GPS attivo. Google utilizza per esempio questa funzionalità con il suo servizio web based Latitude per condividere la posizione dell'utente con i propri amici.

Figura 1.8

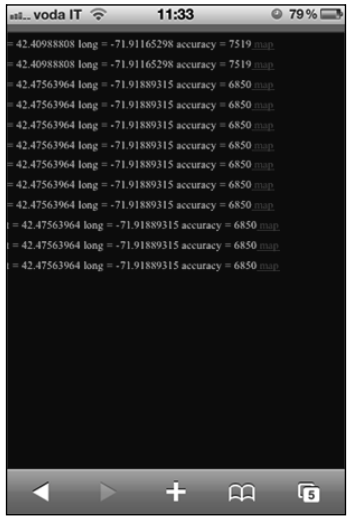

Figura 1.9

- Media tag HTML5: Safari su iOS supporta gli elementi HTML5 `video` e `audio` per incorporare contenuti video e audio nelle pagine web. Questi nuovi elementi eliminano la necessità di impiegare scomodi tag `embed` e `object` per inserire elementi multimediali e consentono di utilizzare una potente API JavaScript. Va inoltre ricordato che i device iOS non supportano Flash, pertanto si può sfruttare il tag `video` per aggiungere file QuickTime con estensione `.mov`. Safari è il principale browser con supporto completo dei media tag HTML5; si deve prestare attenzione quando li si utilizza per realizzare siti web standard, dato che altri browser non ne supportano tuttora l'impiego. Nel caso in cui si stia invece realizzando un'applicazione specifica per device iOS, si può fare completo affidamento sull'utilizzo dei media tag.
- Animazione ed effetti CSS: la nuova versione di Safari supporta l'*animazione CSS*, che permette di elaborare gli elementi in molti modi, per esempio tramite funzioni di scala, rotazione, dissolvenza e inclinazione. Safari su iOS supporta anche gli *effetti CSS*, mediante i quali si possono utilizzare comandi CSS per realizzare sfumature, maschere e riflessi.
- SVG: il formato XML SVG (*Scalable Vector Graphics*) consente di costruire oggetti grafici vettoriali di tipo statico e animato. Grazie al supporto del formato SVG, Safari su iOS offre non solo una soluzione per elaborare grafica scalabile, ma fornisce una tecnologia in grado di escludere la necessità di usare Flash per la creazione di elementi multimediali animati.

Quattro soluzioni per lo sviluppo di web app per iOS

Un'applicazione web da eseguire in un browser qualsiasi e una web app iOS nascono sfruttando ingredienti comuni, ovvero HTML, CSS, JavaScript e AJAX, ma non possono essere considerate identiche tra loro. Il lavoro di sviluppo per iOS prevede quattro diversi livelli di approccio.

- Livello 1 – Sito web o applicazione pienamente compatibile: l'approccio di base è lo sviluppo di un sito web oppure una web app di tipo *iOS friendly*, che sia cioè pienamente compatibile con i device mobili di Apple, come mostrato nella Figura 1.10. I siti di questo genere escludono l'impiego di tecnologie che non sono supportate dai device mobili di Apple, come Flash, Java e altri plug-in. La struttura principale del livello di presentazione si basa inoltre massicciamente su blocchi e colonne che semplificano all'utente la navigazione e le funzioni di zoom. Questa tecnica non aggiunge funzionalità specifiche degli utenti iOS, ma si limita a garantire che non ci siano ostacoli a una navigazione efficace.

Figura 1.10

- Livello 2 – Sito web o applicazione ottimizzata per Safari su iOS: il secondo livello di supporto per iOS non garantisce solo un livello di base di navigazione in Safari su iOS, ma fornisce strumenti che ottimizzano le funzionalità degli utenti di browser Safari, per esempio grazie all'adozione di proprietà avanzate di WebKit CSS supportate da Safari.
- Livello 3 – Sito web o applicazione per device mobili: il terzo livello di supporto punta sul fatto che il sito web o la web app vengano confezionati su misura per la vista di iPhone e iPad, allo scopo di predisporre potenti strumenti di navigazione per tutti gli utenti di device mobili, come si può vedere nelle Figure 1.11 e 1.12. I siti sono strutturati in modo da ottimizzare la vista nei device mobili, anche se non sfruttano pienamente le funzionalità proprie del sistema iOS. In molti casi si tratta inoltre di versioni ridotte di un sito o di un'applicazione web ben più complessi.

Figura 1.11

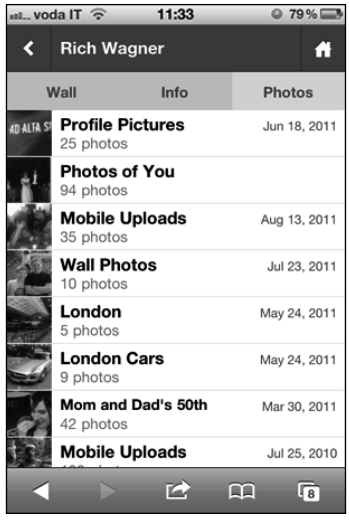

Figura 1.12

- Livello 4 – Applicazione web per la piattaforma iOS: la tecnica conclusiva prevede di sviluppare un'applicazione web progettata esclusivamente per iPhone e iPad, con un design dell'interfaccia utente che richiama quella di un'app nativa o che sfrutti a pieno titolo le funzionalità dei device iOS, come mostrato nella Figura 1.13. Uno degli obiettivi di progetto è ridurre al minimo la consapevolezza da parte dell'utente di trovarsi nell'ambiente di navigazione di un browser. Va inoltre ricordato che un'applicazione iOS di questo genere si integra con i servizi specifici del sistema iOS, comprese le app Telefono Mail e Mappe.

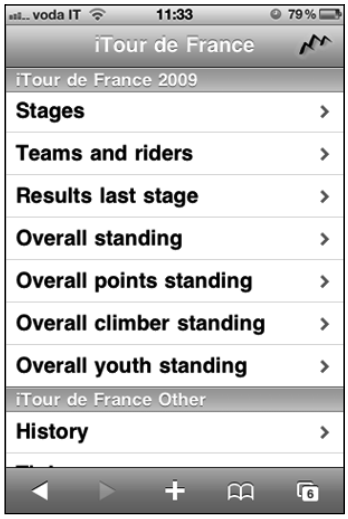

Figura 1.13

In definitiva, le specifiche di una determinata applicazione identificano il livello di utilizzo che si vuole offrire agli utenti iOS e si deve progettare l'applicazione tenendo conto del livello adottato. In questo libro si farà riferimento fondamentalmente allo sviluppo di applicazioni web ottimizzate per il sistema iOS.

Il dito non è un mouse

Una delle considerazioni fondamentali da tenere ben presente quando si sviluppano applicazioni per iOS è che il dito non è un mouse. I computer desktop permettono l'utilizzo di svariati dispositivi di input, per esempio un Apple Magic Mouse, un trackball Logitech oppure il touchpad di un portatile. In ogni caso il puntatore del mouse assume sullo schermo sempre la medesima forma, dimensione e ha sempre lo stesso comportamento. Nei sistemi iOS il dispositivo di puntamento tende invece ad assumere una forma particolare ed esclusiva; questo perché, per esempio, è probabile che le danzatrici abbiano dita piccole e sottili, mentre i giocatori di rugby hanno dita robuste e voluminose, anche se la maggior parte degli utenti si trova in una condizione intermedia tra questi due casi limite. Va inoltre ricordato che le dita non sono precise come i puntatori dei mouse: le dimensione dell'interfaccia e le funzioni di posizionamento del cursore diventano quindi molto importanti, a prescindere dal fatto di creare un sito web *iOS friendly* oppure un'applicazione web sviluppata per iPhone o per iPad.
L'input effettuato con le dita non corrisponde sempre all'input con il mouse. Un mouse permette il clic sinistro, destro, lo scorrimento e lo spostamento. Al contrario, il dito consente gesti come il toccare (*tap*), lo strisciare (*flick*), il trascinare (*drag*) e il pizzicare (*pinch*). Chi sviluppa applicazioni deve tenere conto dei gesti che l'applicazione è in grado di supportare. Alcuni gesti tipici della navigazione dei siti web (per esempio lo zoom con il doppio tocco) non vengono in genere inclusi tra quelli supportati da un'applicazione iOS. La Tabella 1.2 riporta i gesti supportati dal sistema iOS con l'indicazione se devono essere inclusi nella navigazione di un sito web oppure di una web app.

Tabella 1.2 I gesti delle dita.

Gesto	Risultato	Sito web	App
Toccare (*tap*).	Equivale a un clic del mouse.	Sì	Sì
Trascinare (*drag*).	Permette di spostarsi nella vista.	Sì	Sì
Strisciare (*flick*).	Fa scorrere una pagina o un elenco in su o in giù.	Sì	Sì
Doppio tocco (*double tap*).	Ingrandisce e centra un blocco di contenuti.	Sì	No
Aprire allontanando due dita (*pinch open*).	Ingrandisce il contenuto.	Sì	No
Chiudere avvicinando due dita (*pinch close*).	Rimpicciolisce il contenuto per visualizzare una parte più ampia della pagina.	Sì	No
Toccare e tenere premuto.	Visualizza un messaggio informativo.	Sì	No
Scorrere con due dita.	Fa scorrere in su o in giù un iframe o un elemento che ha la proprietà CSS `overflow:auto`.	Sì	Sì

Limitazioni e vincoli

Dato che iPhone, iPad e iPod sono device mobili, quando si sviluppano applicazioni si deve tenere conto delle ovvie risorse limitate a disposizione. La Tabella 1.3 elenca i limiti delle risorse e i vincoli di carattere tecnico. Va inoltre ricordato che non sono supportate alcune tecnologie, come si può vedere nella Tabella 1.4, ed è perciò necessario evitarle quando si sviluppano applicazioni per device iOS.

Tabella 1.3 Vincoli delle risorse disponibili.

Risorsa	Limitazione
Risorse di testo in download (file HTML, CSS, JavaScript).	10 MB.
Immagini JPEG.	128 MB (le immagini JPEG che occupano più di 2 MB vengono scalate di 16x).
Immagini PNG, GIF e TIFF.	8 MB (in altre parole, larghezza × altezza × 4 < 8 MB).
GIF animate.	Una dimensione minore di 2 MB garantisce il mantenimento del frame rate (se maggiore di 2 MB, si visualizza solo il primo frame).
File multimediali non in streaming.	10 MB.
Documenti PDF, Word, Excel.	30 MB e oltre (molto lento).
Allocazione di stack e oggetti JavaScript.	10 MB.
Limite di esecuzione di JavaScript.	10 secondi per ogni entry point di livello superiore (dopo 10 secondi si chiama la funzione `catch` in un blocco `try/catch`).
Pagine aperte in Mobile Safari.	8 pagine.

Tabella 1.4 Tecnologie non supportate da iPhone e iPod touch.

Area	Tecnologie non supportate
Tecnologie web.	Media Flash (SWF e FLV), applet Java, SOAP, XSLT e installazione di plug-in.
Tecnologie mobili.	WML.
Accesso a file.	Accesso al file system locale.
Sicurezza.	Protocollo Diffie-Hellman, chiavi DSA, certificati self-signed, certificati x.509 personalizzati.
Eventi JavaScript.	Diversi eventi del mouse.
Comandi JavaScript.	`showModalDialog()`
Icone bookmark.	File `.ico`.
HTML.	Input `type="file"`, tooltip.
CSS.	Stili di passaggio del mouse, `position:fixed`.

Impostare l'ambiente di sviluppo in una rete locale

Il sistema iOS non consente l'accesso al file system locale, perciò non è possibile aggiungere direttamente una nuova applicazione in un device iOS. Di conseguenza, è necessario accedere all'applicazione web tramite un altro computer. Nel caso di applicazione *live* si dovrà ovviamente collocare l'applicazione in un server web accessibile da tutti, mentre in fase di test la questione va affrontata in modo diverso. Se disponete di una rete Wi-Fi a casa oppure in ufficio, si suggerisce di configurare un web server nel proprio computer principale da impiegare come server di test in fase di distribuzione dell'applicazione (*deployment*).

Il sistema OS X fornisce un web server Apache già installato nel computer. Per abilitare l'accesso a iOS è sufficiente aprire *Preferenze di Sistema*, accedere alla sezione *Condivisione* e attivare l'opzione *Condivisione web*, come si può vedere nella Figura 1.14. Dopo aver attivato l'opzione, si può vedrà l'URL del proprio sito web nella parte inferiore della finestra; questo URL di base consente a un device iOS di accedere ai file web.

Potete aggiungere file nella directory del sito web installata nel computer *(/Libreria/WebServer/Documents*) oppure nella directory del sito web personale *(/Utenti/*VostroNome*/Siti*) e poi accedere ai file inclusi utilizzando la barra degli URL di iPhone o iPad, come nell'esempio mostrato nella Figura 1.15.

Esercizi

1. Qual è la differenza tra un'app iOS nativa e una web app?
2. È possibile collocare nella schermata *Home* una web app a fianco di app native?
3. Il tocco con le dita corrisponde sempre all'input del mouse?

Trovate le risposte nell'Appendice.

Figura 1.14

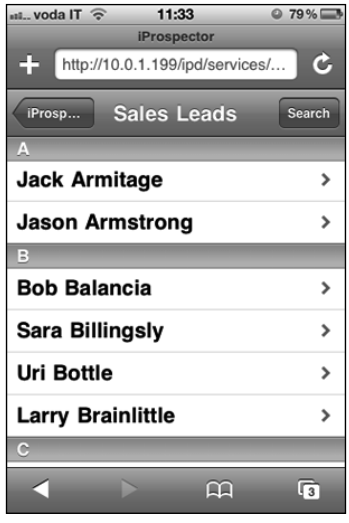

Figura 1.15

Cosa avete imparato in questo capitolo

Argomento	Concetti fondamentali
Funzioni di Safari fondamentali per gli sviluppatori.	Supporto della geolocalizzazione, dei media tag HTML5, di animazioni ed effetti CSS e del formato SVG.
Supporto dei device mobili da parte del sito web.	Livello 1: applicazione/sito web pienamente compatibile. Livello 2: applicazione/sito web ottimizzata per Safari su iOS. Livello 3: applicazione/sito web per device mobili. Livello 4: applicazione web per la piattaforma iOS.