



Cordova e Phonegap: applicazioni mobili per tutti gli ambienti

Sviluppare per diverse piattaforme cellulari con gli strumenti del web.

Di recente Molly Holzschlag, una dei guru di Html e Css, ha chiesto su Twitter di definire Html5 in una parola o in un tweet. La risposta che ci è venuta in mente d'acchito è: un linguaggio per la descrizione di interfacce utente, o quasi.

Html era nato molti anni fa come linguaggio per la descrizione di documenti, una derivazione del più austero Sgml. Dopo pochi anni, la disponibilità di un sistema potente per la creazione di decorazioni e stili di visualizzazione, Css, e la disponibilità di un linguaggio di programmazione bizzarro, ma efficace, JavaScript, hanno reso sempre più popolari le applicazioni web.

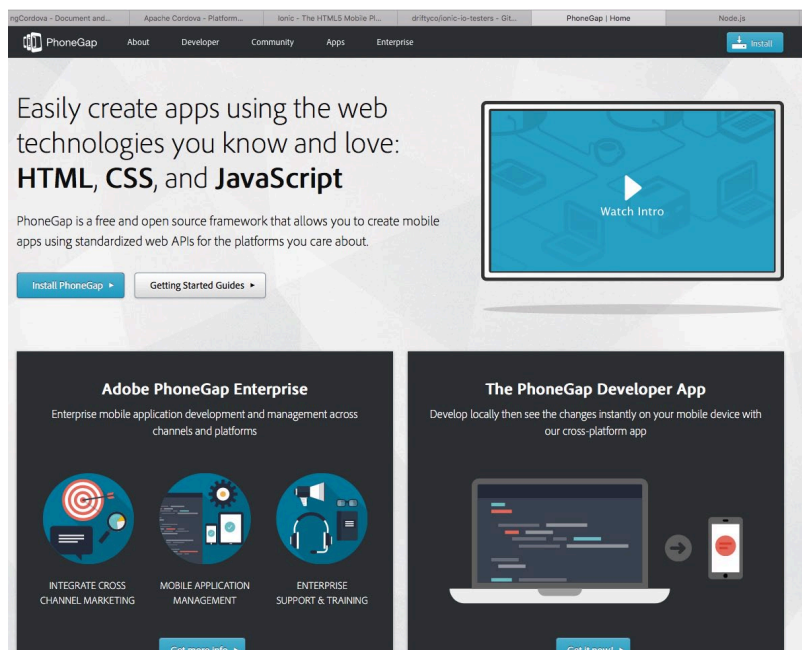
La prima risposta del Web Consortium, Xhtml, è stata un po' una negazione di questa idea, un tentativo di pilotare Html verso un mondo più puro e astratto. Facendone un linguaggio strettamente specificato per la descrizione di documenti, ma negli stessi anni si è imposto Html5, una specifica che rappresenta la ratificazione del fatto che

Html può essere usato come linguaggio per la descrizione di applicazioni web con interazioni complesse.

Nel panorama di oggi, Html5 è a pieno titolo una piattaforma applicativa, che sta sbarcando sui cellulari portandosi dietro la sua semplicità di sviluppo e milioni di sviluppatori. Questo mese

parliamo di due piattaforme, strettamente imparentate che chiudono il gap fra lo sviluppo nel browser e lo sviluppo nativo: Cordova e Phonegap, lasciando fuori dal quadro, per il momento, Ionic, che è una piattaforma interessante e di cui ci occuperemo, che richiede più spazio.

Il sito web di Phonegap è ricco di documentazione e di risorse, come l'applicazione di sviluppo e la app di test per il telefono



Node.js è un ambiente estremamente fecondo, che ha dato origine ai progetti più disparati provando ogni volta le sue performance. Il sistema di distribuzione di pacchetti *npm* è molto semplice e efficace.

Questi ambienti ci permettono di realizzare applicazioni nuove usando tool e linguaggi familiari, sia pure con estensioni nuove. Un editor di testi, Html5 e Css e un po' di studio della piattaforma ci danno la possibilità di entrare nello sviluppo mobile armati delle conoscenze nello sviluppo web.

WEB APP

Ragioniamo, innanzi tutto, in termini architetturali mettendo a fuoco la differenza fra una applicazione nativa e una web app.

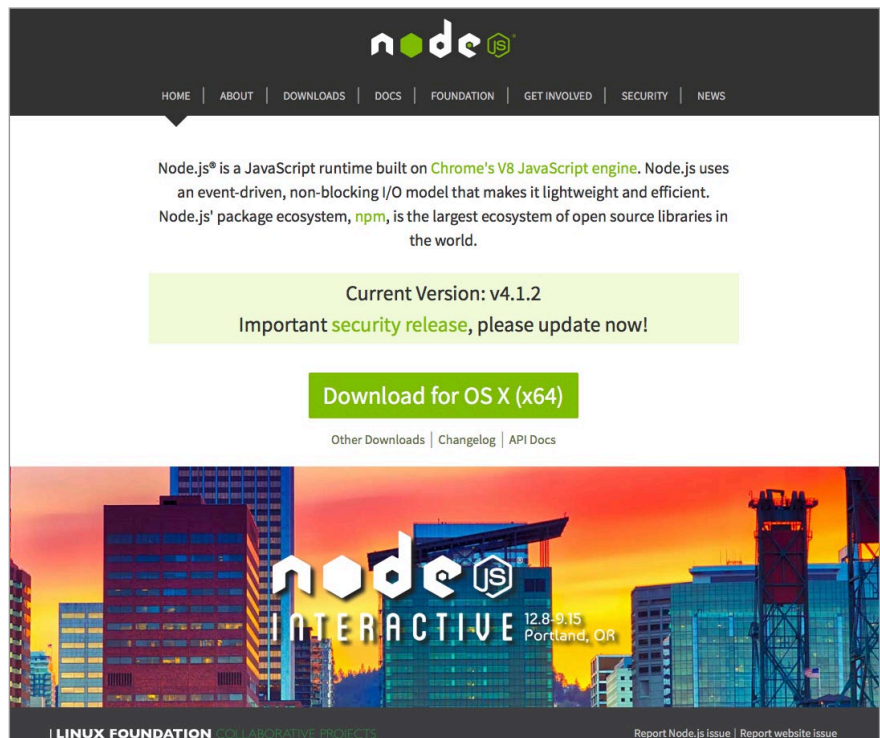
Le applicazioni native per ogni piattaforma sono quelle realizzate con gli strumenti e il workflow dettato dalla piattaforma, per esempio applicazioni Objective-C o Swift su iOS e applicazioni Java su Android. Per web app, invece, si intende una applicazione generata con Html (Html5), Css (Css3) e JavaScript, gli strumenti tipici del web, pacchettizzata in modo da essere sperimentata come un'applicazione nativa, cioè venduta o distribuita sugli app store e con pieno accesso alle funzioni avanzate dell'hardware del dispositivo. Una web app, concettualmente, è formata da tre strati. Il primo è un layer di supporto, simile a quello di un'applicazione creata con i tool tipici della piattaforma. Il secondo è uno strato di interfaccia, scritto in codice nativo,

che esporta le funzioni hardware del telefono, come fotocamera, accelerometro, bussola e le strutture dati, come i contatti e il calendario. L'ultimo è lo strato superiore, scritto in Html e Css, come un'applicazione da eseguire nel browser, ma facendo uso delle funzioni native del telefono in modo diverso da come si fa sul telefono.

Uno studio di Forrester Research ricorda che le applicazioni client server sono state inizialmente realizzate con client nativi, ma sono passate al browser nel corso del tempo e pronostica un percorso simile per le applicazioni mobili, ma le applicazioni web vengono da un periodo di rigetto. La critica più famosa l'ha fatta Zuckerberg in persona, affermando in un'intervista che scegliere un

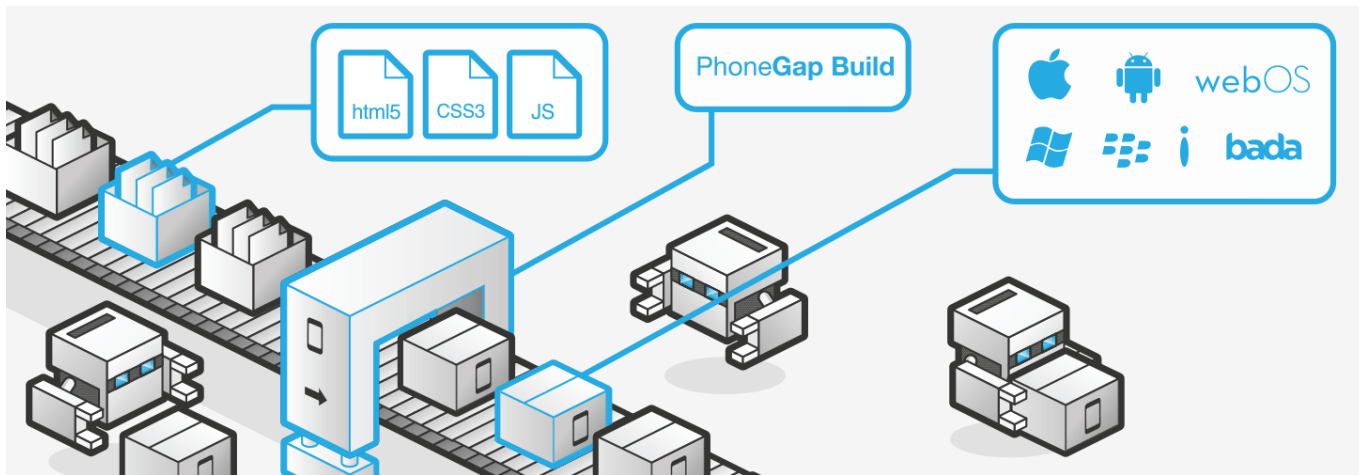
approccio web per l'applicazione è stato uno degli errori principali di Facebook. Qualcosa è cambiato e, qualche anno, dopo, Facebook è alacremente al lavoro su una piattaforma JavaScript, chiamata React.js (insieme a altri progetti su facebook.github.com).

Una forza alla base del ritorno delle applicazioni web è un progetto Apache, quindi open source, chiamato Cordova, creato con Node.js (nodejs.org), un ambiente di esecuzione che contiene il motore JavaScript di Google Chrome. Per la sua natura di applicazione nativa che contiene un interprete, Node può offrire un ambiente di esecuzione simile al browser, ma in grado di esportare al codice JavaScript funzionalità di base che il browser, per sicurezza, nasconde.



LE FUNZIONALITÀ HARDWARE SUPPORTATE DA CORDOVA/PHONEGAP

Feature	iPhone / iPhone 3G	iPhone da 3GS in poi	Android 1.0 - 4.4	Windows Phone	BlackBerry 10 e PlayBook OS	BlackBerry OS 4.6-4.7	BlackBerry OS 5.0-6.0+	Bada	Symbian
Accelerometro	●	●	●	●	●	×	●	●	●
Fotocamera	●	●	●	●	●	×	●	●	●
Bussola	×	●	●	●	●	×	×	●	×
Contatti	●	●	●	●	●	×	●	●	●
File	●	●	●	●	●	×	●	×	×
Geolocalizzazione	●	●	●	●	●	●	●	●	●
Media	●	●	●	●	●	×	×	×	×
Rete	●	●	●	●	●	●	●	●	●
Notifiche (alert, suono, vibrazione)	●	●	●	●	●	●	●	●	●
Memorizzazione	●	●	●	●	●	×	●	×	●



Il sistema di compilazione cloud Adobe permette di inviare i propri file Html, JavaScript e Css e ricevere i pacchetti distribuibili per gli OS supportati.

Node è stato corredato di un sistema per la gestione di pacchetti aggiuntivi, chiamato npm (Node Package Manager), che ha contribuito non poco alla sua diffusione, risolvendo la complessità di gestione di un ambiente molto modulare e frastagliato in modo simile a quello in cui è stata risolta la pacchettizzazione del software su Linux.

L'architettura, la modularità, la facilità di pacchettizzazione di Node hanno stuzzicato la creatività, dando origine a numerosi progetti lato server e lato client. Il più interessante, riguardo al nostro tema, è Apache Cordova.

CORDOVA

Si tratta di un progetto Apache, distribuito con licenza open source, che è stato creato da un'azienda canadese, Nitobi, e donato alla Apache Software Foundation. Cordova permette di creare applicazioni mobili multiplatforma con una singola base di codice JavaScript. L'ambiente di esecuzione mette

a disposizione dello sviluppatore tutte le funzioni specifiche dell'hardware del telefono e consente l'estensione per mezzo di plugin.

L'installazione si fa, ovviamente con npm, quindi, per iniziare occorre scaricare il pacchetto contenente node e npm da nodejs.org. Il software è disponibile per Windows, OS X e Linux.

Dalla riga di comando di Windows o OS X digitiamo:

```
npm install -g cordova
```

per installare l'ambiente Cordova. Il flag `-g` indica che vogliamo un'installazione globale, cioè per tutti gli utenti. Questo richiede i permessi di amministratore sulla macchina su cui operiamo. Se abbiamo configurato il nostro utente come utente non privilegiato, come è buona pratica, dovremo autorizzare l'installazione, o premettere la parola chiave `sudo` ai comandi su Linux o OS X. In caso di problemi bisognerà

abilitare la nostra login all'uso di sudo, per esempio, su Mac, collegandosi con una login di amministrazione e aggiungendo la riga

```
nomeutente ALL=(ALL) ALL
```

al file `/private/etc/sudoers`.

Per creare lo scheletro di un'applicazione, lanciamo il comando

```
cordova create MyApp
cd MyApp
```

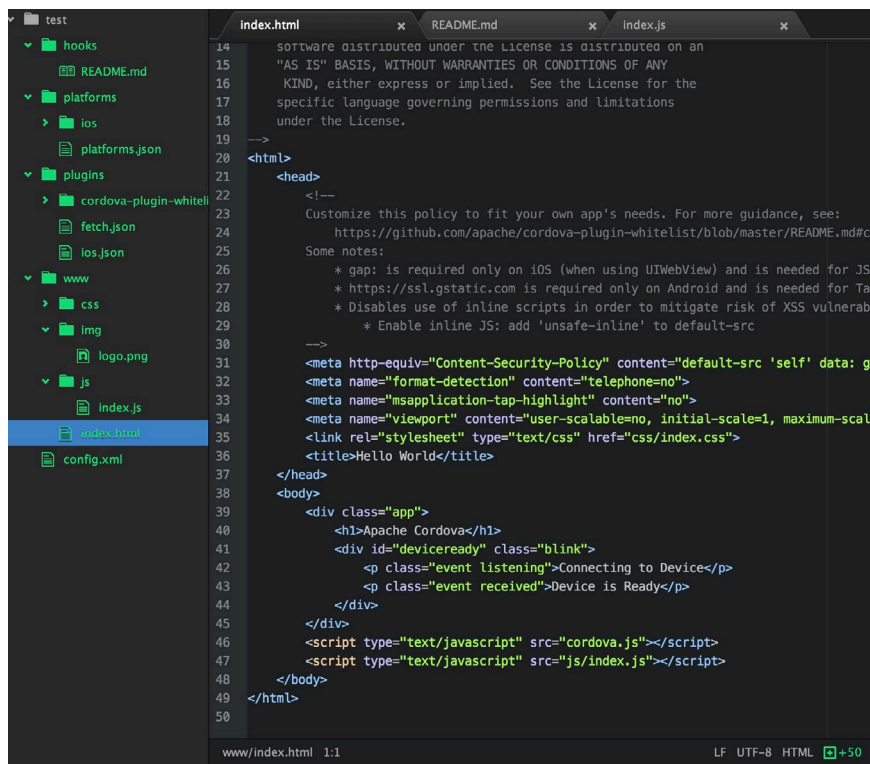
Nella directory MyApp troveremo le cartelle `www`, `plugins` e `platforms`. La prima contiene i file `html` e `JavaScript` dell'applicazione, la seconda i plugin Cordova per le funzionalità aggiuntive e l'ultima contiene il codice per ogni piattaforma.

Inizialmente, questa cartella è vuota.
Popoliamola, ad esempio con

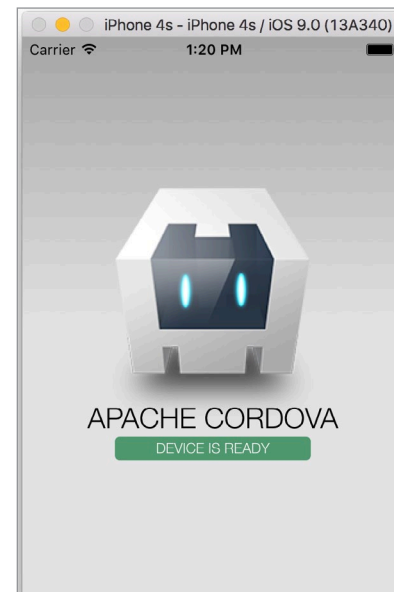
```
cordova platform add ios
```

	webOS	Tizen	Ubuntu Touch	Firefox OS
✓	●	●	●	●
✓	●	●	●	●
✓	●	●	●	●
✗	✗	●	✗	●
✗	✗	●	●	✗
✓	●	●	●	●
✗	✗	●	●	✗
✓	●	●	●	●
✓	●	●	●	●
✓	●	●	●	●

The screenshot shows the Xcode interface. On the left, the 'Project' tab is selected, showing a list of targets: 'iPhoneApp' and 'WatchApp'. The 'iPhoneApp' target is selected. On the right, the 'Info.plist' file is open, showing the 'NSAppTransportSecurity' section. The 'NSAppTransportSecurity' key is set to a dictionary with a 'NSAllowsArbitraryLoads' key set to 'YES'. The 'Project' tab also shows the 'Targets' list, with 'iPhoneApp' selected. The 'Info.plist' file is a dictionary with keys for 'NSAppTransportSecurity' and 'NSAppTransportSecurity'.



Un progetto aperto in Atom, l'editor open source creato da Github, un altro progetto multiplatforma basato su Node.js



L'applicazione di test nell'emulatore di iOS.

Adesso abbiamo il codice di un'applicazione iOS pronta per la compilazione, dentro la cartella `platforms/ios`. Se stiamo usando un mac e abbiamo installato l'ambiente Xcode, possiamo dare il comando

```
cordova compile ios
cordova run ios
```

E avremo davanti a noi l'emulatore e il codice dell'applicazione in esecuzione. Senza entrare nei dettagli, che conviene lasciare alla documentazione del pacchetto, ecco un ritaglio di codice JavaScript che mostra come leggere le coordinate Gps in JavaScript

```
this.addLocation = function(event) {
  event.preventDefault();
  navigator.geolocation.
  getCurrentPosition(
    function(position) {
      alert(position.coords.
      latitude + ', ' + position.coords.
      longitude);
    },
    function() {
      alert('Error getting
      location');
    });
  return false;
};
```

Per fare un altro esempio, se desideriamo ricevere notifiche push da un server remoto occorre installare un plugin

```
cordova plugin add phonegap-plugin-push
```

E gestire le notifiche dal server remoto utilizzando lo scheletro di codice che segue, in cui possiamo utilizzare le variabili indicate nei commenti

```
var push = PushNotification.init(
  ({ "android": { "senderID":
    "12345679"},
    "ios": { "alert": "true",
    "badge": "true", "sound": "true"},
    "windows": { } } });

push.
on('registration', function(data) {
  // data.registrationId
});

push.
on('notification', function(data) {
  // data.message,
  // data.title,
  // data.count,
  // data.sound,
  // data.image,
  // data.additionalData
});
```

```
push.on('error', function(e) {
  // e.message
});
```

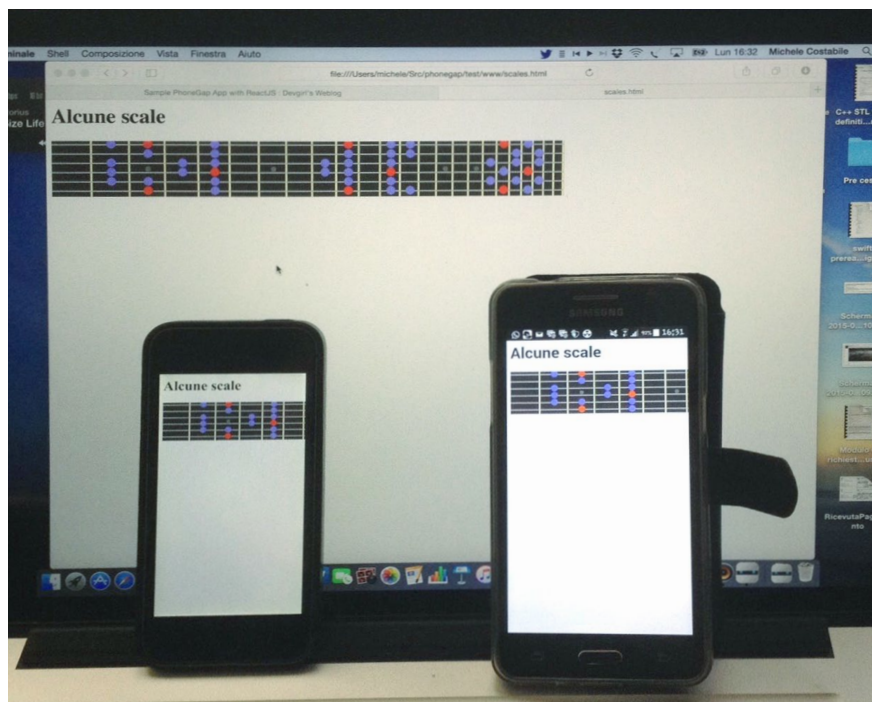
PHONEGAP

Phonegap è il progetto, che ha dato origine a Cordova. Dopo averlo donato alla Apache Software Foundation, Nitobi è andata avanti cercando una posizione di mercato. Nel 2011, Nitobi ha destato l'interesse di Adobe, che l'ha rilevato.

Le differenze fra Phonegap e Cordova sono tutte descrivibili come estensioni. Phonegap offre più funzionalità di Cordova, ma i comandi di Phonegap richiamano quelli di Cordova, nello sviluppo locale e, al massimo può capitare un disallineamento di release fra le due distribuzioni.

Phonegap non è l'unica distribuzione di Cordova con estensioni e una proposta commerciale associata: ci sono anche le proposte di Appery e Ionic, su cui varrà la pena di tornare.

Una delle estensioni più interessanti che Phonegap offre in esclusiva è un servizio per la compilazione di applicazioni native. Grazie a questo servizio, è possibile sviluppare la propria applicazione e collaugarla localmente, senza porsi il problema di avere una



Passare da una pagina web a un'applicazione installata su iOS e Android è questione di poco

catena di compilazione configurata in modo corretto per ogni sistema su cui si vuole distribuire l'app.

I prezzi esposti sul sito build.phonegap.com, sono contenuti.

Esiste un livello gratuito, che consente di lavorare a una singola app, un piano a pagamento da \$9,99 al mese che offre 25 applicazioni e 100MB di spazio e un piano più completo, che offre sempre la possibilità di lavorare su 25 applicazioni e sfruttare 1GB di spazio e richiede un abbonamento a Adobe Creative Cloud.

Phonegap ha sviluppato un'interfaccia grafica verso la gestione dei progetti che permette di fare a meno della riga di comando durante lo sviluppo. Noi non ne abbiamo sentito un gran bisogno, perché i comandi sono tutti facilmente comprensibili e con una sintassi semplice, ma sicuramente l'applicazione semplifica la vita e velocizza il flusso di lavoro.

Phonegap ha un server interno che si lancia dall'applicazione, oppure dalla riga di comando con

phonegap serve

Quando il server è in esecuzione, si può collaudare l'interfaccia di un'app collegandosi con un browser alla porta 3000 del computer su cui è

in esecuzione il server, per esempio 192.168.1.102:3000.

Un'altra esclusiva di Phonegap piuttosto interessante è una app gratuita che si può scaricare dagli App Store di iOS e Android per di collegarsi al server di sviluppo, scaricare l'applicazione e installarla sul telefono per provarla a piena funzionalità, senza le limitazioni del browser.

È un modo facile e veloce di provare l'applicazione in sviluppo, senza la necessità di compilare nativamente e provare l'applicazione attraverso un telefono collegato alla porta Usb del computer. Anche Cordova ha un server interno, che si può mandare in esecuzione con

cordova serve

ma offre un servizio più spartano: l'app si può visualizzare nel browser e non c'è un continuo refresh dell'applicazione che esporta immediatamente ogni modifica.

I due server, comunque possono coesistere tranquillamente. Vale a dire, si possono usare le estensioni di Phonegap anche in un progetto che rimane strettamente Cordova, oppure usa qualche altra distribuzione

commerciale di Cordova. Abbiamo trovato l'app per il telefono preziosa, più che utile, perché il ciclo di sviluppo e test di un'applicazione diventa semplicissimo e si svincola completamente dagli strumenti nativi.

IL VERDETTO

Lavorare con Phonegap/Cordova è veloce e intuitivo per chi ha esperienza di sviluppo web. Possiamo fare leva su tutte le nostre conoscenze in materia di sviluppo html. Il debug si può sempre fare nel browser, con gli strumenti abituali e JavaScript è un linguaggio con le sue stranezze, ma molto facile da usare.

Questo si traduce in un ciclo di sviluppo e test molto veloce. Parlando di collaudo, ci sono diverse piattaforme JavaScript per il test unitario che aiutano a razionalizzare lo sviluppo e toglierlo dalla fase avventurosa.

L'app Phonegap permette di mettere provare un'applicazione sui principali sistemi operativi nel giro di secondi e di vedere immediatamente i risultati delle modifiche, mentre il servizio di compilazione nel cloud toglie la complessità allo sviluppo, eliminando la necessità di una catena di compilazione per ogni target, questo significa che non occorre più di un sistema operativo in laboratorio, dato che i tool di Microsoft e Apple sono legati ai rispettivi sistemi operativi.

Certo, le applicazioni web non sono la soluzione giusta per ogni esigenza. La velocità e l'uniformità di aspetto

con la piattaforma delle applica-

zioni native non si possono discutere, ma applicazioni native, ibride e web hanno punti di forza diversi e hanno ognuna il proprio campo di applicazione.

Infine, le applicazioni web, sono una scelta ragionevole quando si consumano prevalentemente dati provenienti da un server e si vuole un look uniforme su piattaforme diverse. Pensiamo a Facebook o Twitter, per fare un esempio.

La scelta nativa, o ibrida, è più corretta quando si ha a che fare con applicazioni più legate al calcolo, o a funzioni specifiche di una piattaforma, o con giochi in cui un frame rate elevato è importante. •

Nativo o ibrido

La scelta del modello di sviluppo più corretto è complessa: vanno valutate funzionalità e capacità di calcolo.

discutere, ma applicazioni native, ibride e web hanno punti di forza diversi e hanno ognuna il proprio campo di applicazione.

Infine, le applicazioni web, sono una scelta ragionevole quando si consumano prevalentemente dati

provenienti da un server e si vuole un look uniforme su piattaforme diverse. Pensiamo a Facebook o Twitter, per fare un esempio.

La scelta nativa, o ibrida, è più corretta quando si ha a che fare con applicazioni più legate al calcolo, o a funzioni specifiche di una piattaforma, o con giochi in cui un frame rate elevato è importante. •