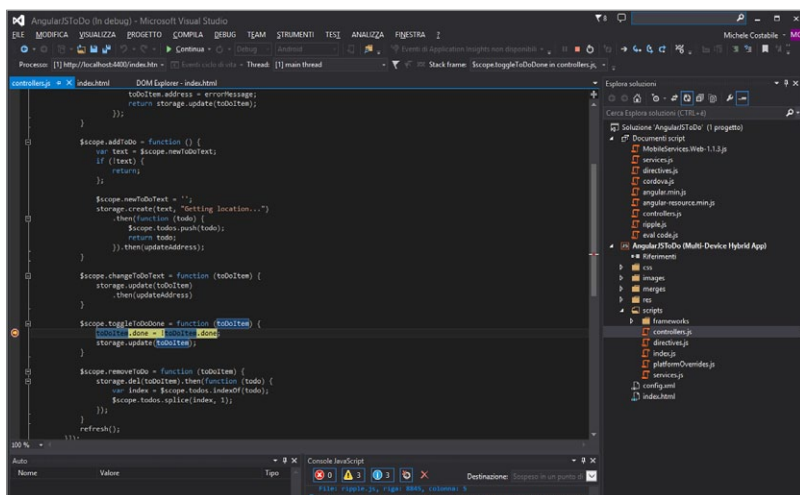


An illustration on a dark teal background. A light-skinned hand holds a black magnifying glass over a computer monitor. The monitor displays a web browser window with a white background and grey borders, containing several lines of stylized code in grey, red, and blue. A large green checkmark is superimposed over the code. A white computer mouse is visible in the bottom left corner.



PC Professionale &gt; Dicembre 2014



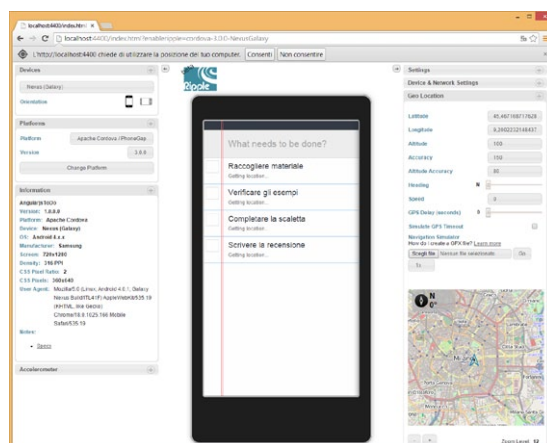
L'integrazione di Cordova con Visual Studio permette di creare breakpoint in JavaScript.

che è facile incontrare quando si prepara un ambiente Cordova su una macchina Unix, come abbiamo provato a fare. Mettere insieme tutti i pezzi giusti del puzzle non è immediato e l'estensione risolve brillantemente il problema di andare a pescare le versioni corrette di Cordova, Node.js, ant e installare tutti i pacchetti con npm. Soprattutto, Microsoft ha integrato l'ambiente di esecuzione con il debugger di Visual Studio, quindi lo sviluppatore può entrare nell'esecuzione di un'applicazione nel modo abituale. L'esecuzione di un'applicazione con diverse piattaforme è facilitata da Ripple, un ambiente di simulazione che può riprodurre le caratteristiche di diversi telefoni Android e degli iPhone, quindi possiamo passare dall'esecuzione simulata su un Nexus 7 a quella su un iPhone 5.

**Naturalmente, è gestito Windows Phone** e le applicazioni sono pacchettizzate automaticamente per la distribuzione sul negozio online di Microsoft. La pacchettizzazione di applicazioni per Android è semplice, richiede solo la generazione di una firma digitale. La pacchettizzazione finale e la distribuzione di un'applicazione iOS richiede un Mac e Xcode (e ovviamente la registrazione al programma per gli sviluppatori di Apple). Le istruzioni per la generazione di pacchetti si trovano su [msdn.microsoft.com/it-it/library/dn757048.aspx](http://msdn.microsoft.com/it-it/library/dn757048.aspx).

Per saperne di più, la pagina iniziale del progetto è [www.visualstudio.com/en-US/explore/cordova-vs](http://www.visualstudio.com/en-US/explore/cordova-vs). Gli sviluppatori possono essere contattati ponendo domande su [StackOverflow.com](http://StackOverflow.com).

L'estensione di Visual Studio contiene Ripple, un emulatore per diverse piattaforme e dimensioni del video.



## XAML, UN LINGUAGGIO PER LE INTERFACCE UTENTE

Questa soluzione è alla base di tutti gli strumenti di sviluppo per le varie versioni di Windows, dal telefono al grande server.

Xaml è indispensabile per creare sia applicazioni Windows con la nuova interfaccia, sia applicazioni per Windows Phone. Sul desktop classico si può evitare di avere a che fare con Xaml se si sceglie di creare applicazioni Windows Forms, mantenendo le abitudini del passato e, eventualmente, il set di controlli creato in casa o acquistato.

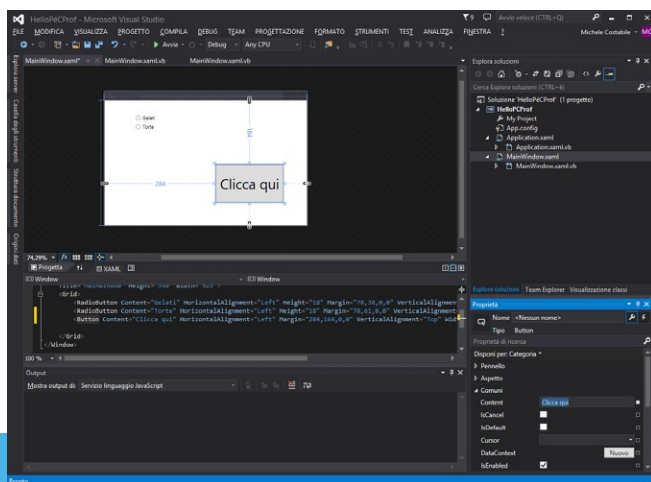
Sta di fatto, però, che Xaml ha occupato tutte le aree di sviluppo possibili per i sistemi operativi di Redmond, lasciando in vita due sacche: l'area legacy in cui continua a sopravvivere Visual Basic 6 e l'area contemporanea delle applicazioni Windows Forms.

Certamente è lecito domandarsi se valga la pena distaccarsi da Windows Forms, per chi ci sta lavorando, sulla base di dubbi sul suo futuro. Non è facile rispondere a questa domanda. Mantenere in vita due ambienti diversi e sovrapposti non è il massimo, ma Microsoft lo ha fatto parecchie volte, anche se è evidente che non ci sono tracce di Windows Forms su telefoni e tablet, come li intende Redmond.

Per chi viene da applicazioni legacy, invece, Windows Forms è certamente un ambiente più familiare, ma (in un numero precedente abbiamo affrontato la questione) un salto generazionale c'è in ogni caso, quindi tanto varrebbe scegliere il linguaggio e la piattaforma applicativa mainstream, quelli su cui si basano gli esempi, gli articoli, il materiale didattico. Intendiamo dire C# e Xaml. La ricompensa immediata è la possibilità di lavorare per telefoni e tablet, anche quelli con processore Arm, la ricompensa finale è, presumibilmente, imparare l'ambiente con vita più lunga.

### CHE COSA È XAML

È un linguaggio dichiarativo per la descrizione di interfacce utente. Dichiarativo, nel senso che non contiene un elenco di istruzioni che servono a creare una pagina, ma una descrizione del contenuto della pagina. In questo è molto simile a Html, con cui condivide



In fase di progettazione di un'interfaccia utente, Visual Studio ci presenta il testo della descrizione dell'interfaccia, in Xaml, insieme all'aspetto dell'applicazione.

## > segue

anche l'aspetto. Xaml, infatti, è un dialetto Xml. Quando ne abbiamo parlato la prima volta, nel numero 196 di PC Professionale, era il 2007 e Xaml compariva come il formato di Silverlight, un formato destinato a ospitare oggetti video e animazioni per quello che Microsoft ha creato per competere con Adobe Flash. Trentadue numeri dopo, su PC Professionale numero 228, Xaml era di nuovo ospite nelle nostre pagine come la base per Wpf, Windows Presentation Foundation, il primo tentativo di Microsoft di razionalizzare la sua tecnologia per la presentazione dei dati e modernizzare l'aspetto dei programmi, facendo leva sui designer con esperienza di Silverlight, in un'epoca in cui era considerato probabile che Html prendesse il sopravvento sugli altri mezzi per la creazione di interfacce utente.

**Oggi, Xaml è cresciuto ed è semplicemente il cuore della presentazione dati** su Windows e Windows Phone, poiché è abbastanza espressivo da poter rappresentare non solo la disposizione dei controlli e i dettagli di eventuali animazioni, ma anche buona parte dell'aspetto tecnico dell'interazione con l'utente, come la validazione dei dati, la paginazione, la creazione di interfacce master/detail.

Questi ultimi aspetti lo posizionano un passo avanti, rispetto a Windows Forms.

Un altro aspetto importante di Xaml, è che l'architettura prevede la possibilità di definire stili di visualizzazione dei controlli, per definire temi di colore o di passare un'applicazione da un look 3d a uno piatto con controlli globali, proprio come è abituale nel web con i fogli di stile. Ci si potrebbe domandare, quindi, come mai Microsoft non abbia adottato il modello di Html e Css, già assestato e definito. La risposta sta nel fatto che in Xaml trovano posto

anche le specifiche dell'interazione con l'utente, del modo in cui si propagano gli eventi nell'applicazione, una scelta fortemente orientata alla descrizione delle applicazioni desktop. Questo caratterizza Xaml in modo specifico ed è risolto con più libertà in una struttura di linguaggio dedicata specificatamente alla creazione di app.

Un altro aspetto da tener presente, è che Microsoft fornisce uno strumento, Blend, orientato al disegno visuale e alla creazione di animazioni e effetti. Blend lavora con file Xaml e può condividere un progetto con Visual Studio. Questo permette di dare a figure diverse lo strumento adeguato, oppure consente allo sviluppatore di passare da un modo all'altro di vedere le stesse cose, dando più peso alla creazione visuale o alla struttura dell'applicazione.

## L'ARCHITETTURA

Il progetto di un'applicazione conterrà un file Xaml per ogni maschera utente o per ogni controllo riutilizzato all'interno dell'applicazione. A fianco di ogni file Xaml troveremo un file corrispondente nel linguaggio scelto, C# o Visual basic.net. C'è anche un file Xaml globale per l'applicazione, anche questo con un file di programma associato, in cui si trovano inizializzazioni globali dell'applicazione, sia per quanto riguarda il codice, sia per ciò che concerne lo stile della presentazione.

Come avviene in Html e Css, si possono riutilizzare parti di codice, descrivendo stili di presentazione e modelli di interazione in un file e riutilizzandoli per tutta l'applicazione. Chiudiamo questa introduzione con qualche minimo esempio di codice, per esempio, lo stretto necessario per definire un pulsante:

```
<Button></Button>
```

Che possiamo espandere minimamente così

```
<Button>Cliccare qui</Button>
```

Possiamo, però anche creare un pulsante che contiene un campo di testo e un secondo pulsante allineati in una griglia. Qui stiamo già divagando, ma l'esercizio ci sembra interessante:

```
<Button>
  <Grid>
    <TextBlock>Ciao a tutti</TextBlock>
    <Button>Clicca qui</Button>
  </Grid>
</Button>
```

In un prossimo numero di PC Professionale vedremo più in dettaglio il processo di creazione di un'interfaccia, passo per passo, e i fondamentali e cardinali agganci sia fra interazione e codice, sia tra interfaccia utente e dati utilizzati all'interno del programma.