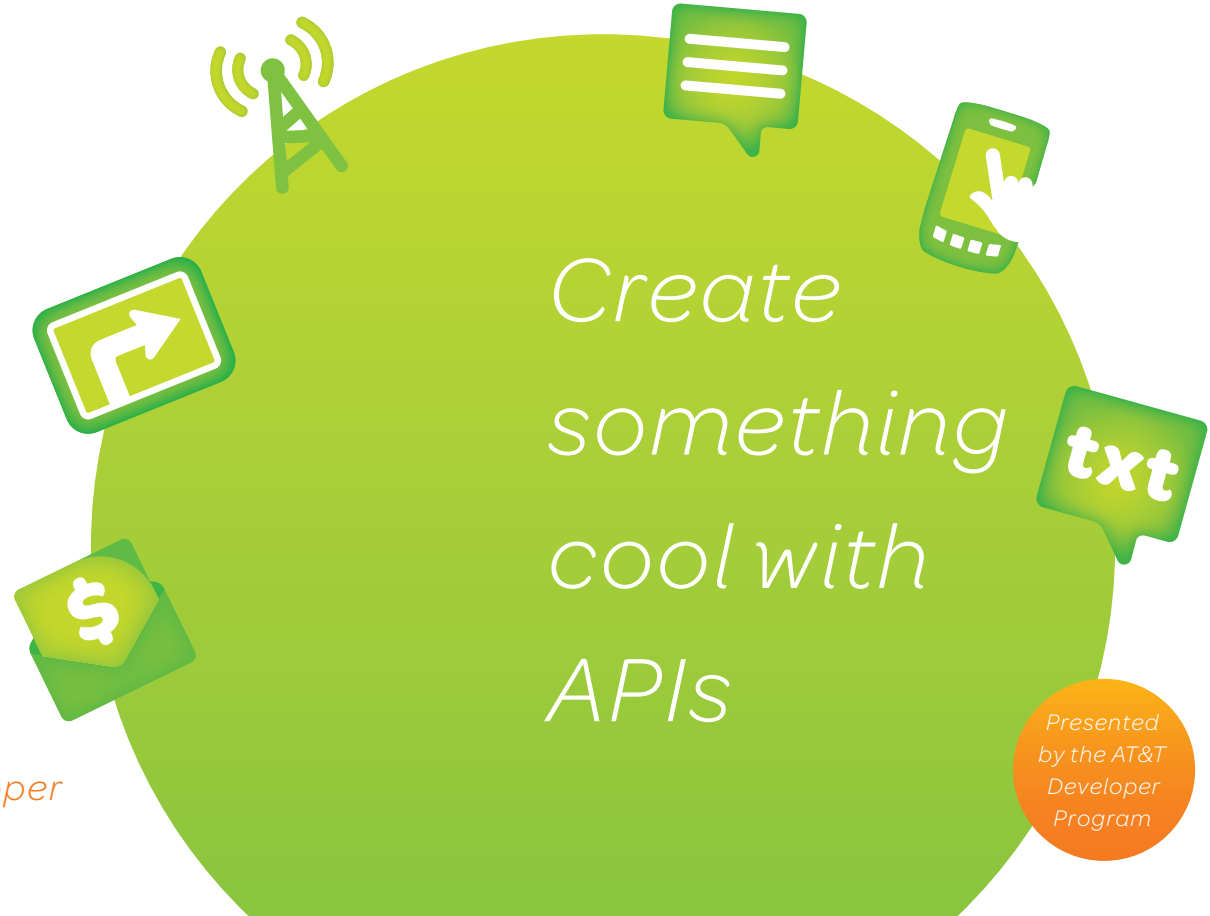
The Developer's Guide to APIs



Create something cool with APIs

Presented by the AT&T Developer Program

developer.att.com
AT&T Developer Program

The Developer's Guide to APIs


Chapter 1
The World
of APIs

Chances are you've used an API before, even if you don't know it. Have you ever used the "Login via Twitter™" button in an app? Received a push notification? Looked at a map within an app? If so, you've used an API! Of course, developing apps that utilize APIs is a little more complex, but you should be starting to get the idea of just what APIs can bring to the table in your mobile app. They open up a wide, wide world of possibilities to bring a lot of other companies' content and services into your app, and, importantly, provide a way for you to potentially make money from your app.

What is an API?

API stands for **A**pplication **P**rogramming **I**nterface. APIs provide a structured way for apps and services to communicate with each other and exchange information. APIs define how requests for services are made, and specify how they will deliver the results of

those requests. APIs allow for application-to-application communication, hiding a lot of complexity from the end user, and allow for data to be presented in a clean, well-designed and useful way.

In layman's terms, APIs are how different apps and services talk to each other. Your mobile app might want to ask the Google Maps™ API for some location information, and the API lets it do that. The API allows your app to make requests and fulfills them, and it behaves in a defined and structured way, so your app knows how to make that request and how the data sent back will be delivered.

What can I do with an API?

Your options are almost limitless! If a provider wants to offer a service to external developers, such as mapping or other location services, they'll provide an API that developers can access and build apps and services that communicate with it. There are more than 8,400 APIs out there for developers to use, according to the Programmable Web® directory as of January 2013. They range from common ones, like the Facebook® API that allows users to log in to apps with their Facebook credentials, to the Twitter API that allows app users to post their activities and actions to their Twitter stream. There are lots of specialized ones too, like the Walgreens® API that lets developers send photos to its stores for printing from their apps, or the AT&T Device Capabilities API,

which lets a developer know what device a user has and the functionality it contains.

Chapters 4 and 5 of this book will give you examples of different categories of APIs, the APIs AT&T provides, and how you can use them. In Chapter 6 you'll find some ideas of some apps you might build using APIs, and how you can mash them up—or combine multiple APIs in an app—to create even more exciting features.

Types of APIs: Device APIs vs. External APIs vs. Network APIs

Within the realm of mobile development, there are three types of APIs. Typically, most developers are initially familiar with device APIs that allow access to device features like the camera or address book. But we will focus this booklet on external and network APIs, which you would use to access web services or services from a mobile carrier. Still, it's good to understand the different types of APIs:

- Device APIs are provided by a platform or mobile device vendor, and allow access to device features and functionality, such as the APIs for the Android™ platform.
- External APIs are provided by a web site or content/service provider who wants to allow outside access to their content, information or service. Examples of this would be the Twitter API or the NPR® API.
- Network APIs are a specific subset of external APIs offered by mobile network operators or carriers

that let developers access unique, network-based services. This includes the APIs available on the AT&T API Platform (see Chapter 5 for all of the AT&T APIs).

How do APIs work?

There are a different technical implementations of APIs, as well as different services and features they can use. Chapters 2 and 3 will help you to understand these, but in general, working with APIs follows these steps:

1. **An app or service makes a request to an API.** For example, a mobile app user enters the dates and locations for a flight they want to take , and the app sends this to a travel service's API.
2. **The API receives the request, and routes it to the server-side application that can process the request.** In our example, the travel service API receives the request from the app, and routes the data to its internal flight lookup application.
3. **The application processes the request and formulates a response based on the business logic of the application.** In our example, the internal application finds matching itineraries for the flight and encapsulates them into a response.
4. **The response is sent back to the requesting app, where it is received and formatted for the user.** In our case, the response and its data are sent back to the

user's app, where that user is presented with a nicely designed screen of flight options and prices.

Who owns the APIs and how do I access them?

APIs are owned by their providers, but they allow you access to the data, functions and content they provide, as long as you abide by their terms and conditions. At a minimum, you'll need to register for access and authenticate when you or your users access it. Some APIs are free, others cost money. Some start free and then fees are added when your usage passes a certain level; it just depends on the provider and their business model. It's important to understand how this will impact your app, so you or your customer don't run up high bills for API transactions!


Chapter 2
Types of
APIs and Examples

There are APIs for all sorts of uses and applications. As mentioned, the Programmable Web directory lists over 8,400 APIs are available to developers as of January 2013, and that doesn't even include the thousands of APIs that companies expose privately. Not all of these are intended specifically for mobile apps and services, but chances are there's an API out there for what you want to do!

In this chapter, we'll go over some of the most popular and common categories of APIs and provide examples of the services and features they can bring to your app:

- Advertising and Analytics
- Content
- Location and Mapping
- Login, Identity and Profile Services
- Social Networking
- Messaging
- Payments and Commerce.

Advertising and Analytics

Advertising on the Web is very mature and well known, as are its leading providers. The mobile advertising market is still relatively young and is working through the early stages of growth, investment and consolidation, with hundreds of providers worldwide ready to deliver ads into your app through their APIs. Typically, mobile advertising companies provide an SDK to ease the integration of their services into your apps.

The mobile ad networks come in a lot of different flavors, but can generally be classified along a few different metrics.

- Blind vs. premium placement—in blind networks, the advertiser doesn't have specific control or choice over where their ad appears; in premium networks, they choose locations specifically.
- Geographic coverage.
- Pricing method—cost per click (CPC), cost per 1000 impressions (CPM), or cost per acquisition (CPA).
- Type of ads served—banners, rich media, video, and others.

What do the advertising APIs do? They help you make money! You can use the advertising APIs to capitalize on your audience and generate revenue. A lot of advertising companies also combine the ad side of things with user-acquisition services for app devel-

opers and can be a useful tool in attracting users to your apps and increasing user engagement.

Mobile analytics services provide APIs and SDKs that help you do a number of things within your app:

- Track user behavior, much like an analytics service for a web site does. Once a user has downloaded your app, you'll want to know how they use it, as well as optimize your monetization or conversion funnels.
- Track bugs, so you can make sure that when your app crashes for a user, you can understand why and fix it!
- Grow your understanding of your user base, so you can acquire users more easily and effectively, provide relevant updates for your app, find cross-promotion partners and most importantly, help you provide the best experience for your customers.

Content

Content APIs are used by news, information, and other content providers to allow access to some, or all, of the providers' content. Typically, they will allow developers to only use a small part of their content for free, in order to generate either traffic back to their own site or related properties, or they'll require developers to display the provider's own advertising in the content they share. For full access, they often require developers to pay a fee.

Also included in this category are other providers who use APIs to provide open access to data and information. This could include retailers, job listing sites, public services, data set providers and more. There is an almost endless supply of data and content out there for you to use in your apps!

Location and Mapping Services

Location and mapping APIs might be the most well known type of APIs that will be mentioned here, thanks to their ubiquity in mobile apps. Many mobile apps feature built-in maps or another type of location information, and there have traditionally been a lot of companies supplying APIs in this space. For some time, Google Maps provided the most popular API not only because of the API's features and the fact that it's a common mapping application on smartphones, but also because it was free. However, after Google changed its pricing model in the last couple of years, offerings from Open Street Maps, Nokia®, Amazon, deCarta® and others have grown in popularity.

You might be thinking: "if a device has GPS capabilities, why do I need a location API?" The GPS sensor in a mobile device provides GPS coordinates to an app, and perhaps direction and speed information; but to get a map, turn an address into coordinates, or get directions, you need a mapping API! In addition, some devices still don't have GPS, and there are network location APIs that can be used to provide

their location to apps and services.

When it comes to mobile apps, the sky is really the limit with how location and mapping services can add value and new features. They can:

- Give users directions and navigation
- Highlight points of interest and businesses nearby the user
- Help users locate particular businesses (such as a retail chain's store finder)
- Help target advertising to a user's location
- Create "geofences" around a particular area that, when crossed, trigger an action
- Help users find their way on public transit
- And lots, lots more!

However, not all location APIs can enable all of the above features, so it's important to understand the different types of services they offer:

- Positioning—determining a field force worker's location through mobile network data, Wi-Fi data, an address, or social network check-ins.
- Geocoding—being able to turn an address into GPS coordinates and vice-versa.
- Navigation and traffic—the ability to determine a route between two points and display it on a map, potentially including information on traffic activity on the route.

- 2D or 3D maps display—showing a standard map, or a 3D display, based on a search or a position reading.
- Overlays—some APIs let you overlay graphics, pins and other information on top of the map to create a customized display.
- Satellite or street views—displays of satellite imagery or street views of a location.
- Location-based ads—serving relevant ads based on the user's location.
- Point of Interest (POI) databases—access to a database of businesses, landmarks and other points of interest.

With so many different features, it's important to make sure the location API you choose supports what you want to do. You will also want to evaluate a few other factors, including specific platform support (i.e. Android, Windows® Phone, etc.), cross-platform support (i.e. JavaScript or HTML5) and price—is it free, does it bill per call, or by service level? You'll also want to look at the technology used by the API. Will you be using onboard GPS from your

users' devices, or do you need a service that can determine their location by another means?

Login, Identity and Profile Services

A login and identity service is basically a service with a large pool of existing users (such as Twitter, Facebook or Google). By implementing their login and identity APIs, it means users can register for your app or service without having to create a completely new account—the user's credentials from his/her social network serve as the login for your app. Many developers prefer this approach rather than implementing their own identity system because it's quicker and faster for users, meaning they're more likely to use it.

We will describe in more detail in Chapter 4, in the Authorization and Authentication section, how OAuth is a very powerful approach and standard for enabling your app to leverage existing credentials from many other companies, making the user's login experience seamless and simple.

However, there are services in this area that have adopted other approaches to capturing a user's login credentials. For example, access to enterprise services or financial information will utilize two-factor authentication (such as with a SecurID® algorithm or SMS-based identity system). For enterprise apps, login via a Salesforce® ID or Oracle® single sign-on might be more appropriate.

Services in this area can be put into four general buckets:

- Identity services, or essentially the ability to log in to an app or service with a third-party ID (such as Facebook, Google, etc.).
- Authentication services, or the "I am who I say I am" services (such as two-factor authentication for banking services).
- Authorization services, which allow your app to "do" things on behalf of their user, such as post to their Facebook wall.
- Profile services, which provide information about the user, such as name, location, interests and so on, to your app or service.

In short, these APIs offer an easier or better way to implement login and identity services to your apps. But they have some other benefits as well, in particular the social and virality benefits of integrating with a social network's identity platform. This means your users can (depending on the API and social network) be "found" by other users based on this information (i.e. the "find my Facebook friends using this app" feature).

Social Networking

Alongside their login and identity services, social networking APIs provide a wide range of other content and features for your app. On their most basic level, they let users of your apps connect with other users to share information such as user-generated content, status updates, photos, videos and more. But there are a lot of highly complex APIs serving up location-based social content, and content from highly niche networks. Pretty much any social network has an API that lets you access some aspect of their content and service from your app. After all, they're social networks and want to spread the content as widely as they can!

The most basic social networking API integrations let your users share actions or messages from your app to their profile on a social networking site (such as posting a Tweet to Twitter or a status update to Facebook), or pull in contacts from a network. More specialized implementations take content or data from one service and "mash" it up with data from another. For instance, an application using such a mashup might take data from daily deal web sites and combine it with content from Yelp™, to let users see where the companies with deals are located and easy access to their Yelp reviews.

There are also a number of social network API providers geared specifically towards mobile games. These could include APIs enabling multi-player

gaming, social leaderboards, cross-game virtual currency, friend lists, cross-promotion and more.

Messaging

Traditionally, messaging APIs have been pretty simple: they could send, and in some cases, receive SMS messages. Then MMS was added to the mix, but things have become even more complex with the addition of push notifications. Whereas an SMS can only carry a message, push notifications can carry a message and an action, meaning that instead of, say, a text that says "Your flight is cancelled," a push notification could be sent, opening a travel app or taking some other action in addition to the message.

SMS APIs are provided by mobile operators, aggregators, and other over-the-top (OTT) players. There are benefits to working with each. For instance, aggregators can offer connections to users across multiple operator networks, while operators can offer more features and specific benefits. As an example, the AT&T SMS API allocates short codes to developers' applications at no extra charge, and it can also support a Free to End User (FTEU) model where users don't incur a cost for messages to or from an application.

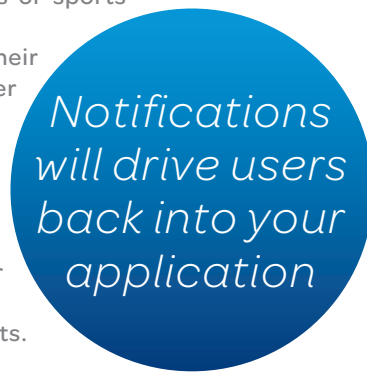
The idea of what constitutes a "message" that can be sent through an API has expanded significantly beyond the simple text message, with specific content

and platform providers offering services that send notifications, or audio or video messages.

Notification and messaging is a hugely important area for developers to think about when building their apps. Notifications will drive users back into your application when they're not using it and let them know something has happened or changed (such as the next turn in a game). Messaging enables person-to-person, or service-to-person messaging that's crucial for unlocking the social potential of your mobile app or service. Messaging APIs let your app live on even when users aren't actively using your app on their phone!

You might use a messaging or notification API to:

- Validate the login credentials of the user
- Send updates of news or sports scores
- Let a user know it's their turn in a multi-player game
- Give status updates of online services
- Send location-based notifications of local points of interest or activity
- Run SMS voting contests.

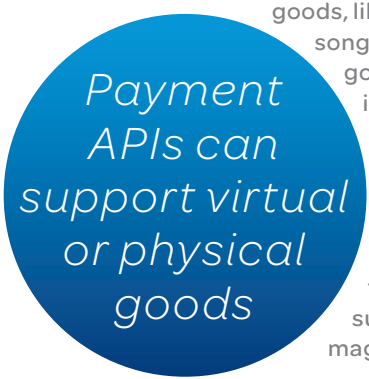


Payments and Commerce

Chances are if you're a developer, you like getting paid. Payments and commerce APIs do just that, supporting a number of different payment models, goods and services, payment methods and geographies.

In-app purchases have now become the leading source of mobile app revenue, accounting for 70 percent or more of the total revenues on the leading platforms, according to the analytics company App Annie™.[1] In order to bill customers and distribute in-app purchases such as new levels in a game, additional features and services, or other items, developers can use APIs from a number of different providers (though sometimes this choice is limited by the terms of a platform or app store).

Payment APIs can support the sale of both virtual goods, like extra levels in a game or song downloads, and physical goods, such as a user ordering a pizza and paying for it within an app. In addition, some will also allow you to use subscription billing, where you bill users on a recurring basis for a product or service, such as a newspaper or magazine subscription.



Not all APIs support all models, so make sure that the one you choose allows what you want to do. You also need to consider the payment methods supported by a particular API; you want to use the methods with the lowest "friction" (meaning they're the easiest for the end user) and the highest conversion rate and best revenue share. Typically, carrier billing, in which a user is able to bill a purchase to their mobile bill, offers higher conversion rates in mobile apps than credit-card billing, simply because fewer clicks are involved (see data from BlackBerry®[2] or Nokia[3]); and many customers don't have credit cards. Other payment methods can include premium SMS, third-party wallets like PayPal™ or Amazon Payments™, or virtual currency, such as Facebook credits.

1) http://gigaom.com/mobile/freemium-app-revenue-growth-leaves-premium-in-the-dust/
2) http://devblog.blackberry.com/2012/11/app-world-integrated-carrier-billing/
3) http://machinsights.wordpress.com/2011/08/08/direct-operator-billing-spreading-the-news/


Chapter 3
Understanding
API Technology

Communicate through a common interface

APIs come in a lot of different flavors of technologies, but REST and SOAP are the most widely used for web-based APIs. Some API providers also offer native SDKs (software development kits) that help mobile app developers integrate their APIs into native apps. In this section, we will talk about some of the technologies used for APIs and how they work.

REST and SOAP were created to allow applications running on different platforms to communicate. What this means is that PCs, servers, tablets and phones can all communicate through a common interface, even though they might be running on completely different platforms.

In the last few years, RESTful APIs have become the most popular choice for developers, because of the simplicity of its implementation over SOAP APIs. With RESTful APIs, developers can choose between the data format they would like to send or receive, and with a set of verbs to use (e.g.: GET, POST), it

makes these APIs more intuitive for developers to use without necessarily looking at the documentation. RESTful APIs also typically use HTTP protocol, making documentation generally easier to understand for developers.

SOAP APIs do offer some advantages, such as the ability to offer mechanisms for success and re-try logic, providing end-to-end reliability through SOAP intermediaries. This is advantageous for companies, like financial institutions, where it is necessary to be certain that operations have completed.

What is REST?

Representational State Transfer is an architecture that was originally introduced and defined by Roy Fielding as an architecture style for the World Wide Web. REST is not a standard or a protocol, it is an architecture that is commonly held as the standard for usable, well-designed, easy-to-integrate APIs. The core concept of a REST architecture is based on a client and server architecture and uses a standard request and response mechanism. The architecture is formed around resources and components whose behavior is encompassed through the verbs GET, POST, PUT and DELETE used for the "CRUD" (create/read/update/delete) operations available to developers.

The RESTful style of API focuses on accessing named resources through a single consistent inter-

face, with most API providers choosing to use the GET and POST verbs for operations related to retrieving data and sending data, respectively. Because of their simplicity and ease of use, RESTful APIs have become the most popular type of API, with many companies like Amazon®, eBay® and Yahoo® now only offering RESTful APIs to developers.

How does a RESTful API work?

RESTful APIs' documentation will specify a resource URL to be used for the operation. Take for example, this URL from the Twitter API, which returns the 20 latest mentions of an authenticated user:

https://api.twitter.com/1.1/statuses/mentions_timeline.json

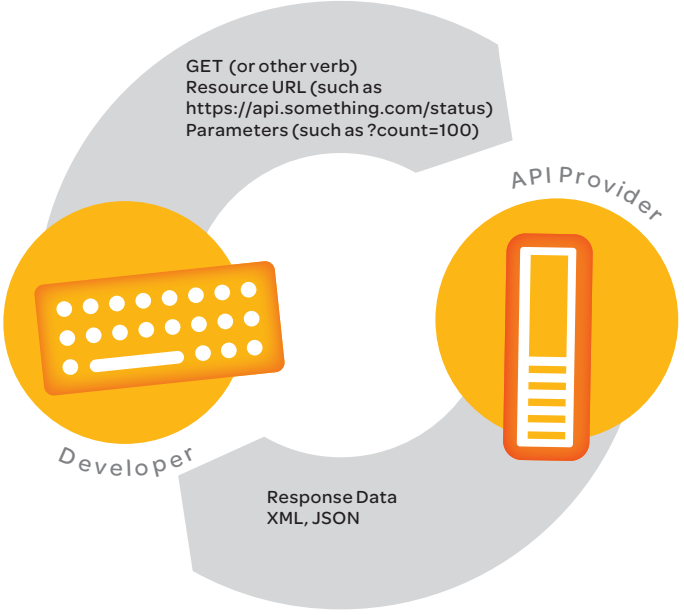
This URL can be augmented with additional parameters that shape the data that is returned, specifying search criteria, limiting the number of responses, and so on. For instance, in the above Twitter example, the "count" parameter specifies the number of Tweets to return. So if you wanted to get the last 100 Tweets (instead of the default 20), the URL in the GET request would be:

https://api.twitter.com/1.1/statuses/mentions_timeline.json?count=100

As you can see, forming requests for RESTful APIs can be pretty easy! This explains a great deal of REST's popularity among developers. When a RESTful API responds to a request with data, it can be sent in any format. JSON is the most widely used (as it can be

easily processed by JavaScript), but XML is also used, and many APIs allow the developer to specify which they'd like to receive in their call.

The diagram below shows how a RESTful API request is structured.



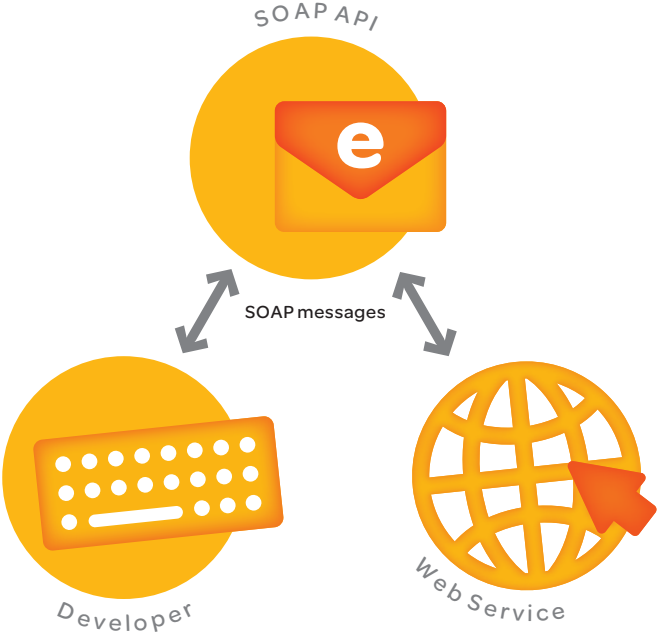
What is SOAP?

Simple **O**bject **A**ccess **P**roto-col is a protocol designed to expose the operations available to a developer and is used for some APIs. SOAP combines HTTP and XML, and since these protocols are known by a wide variety of platforms, it means that computers, servers, phones and tablets can all communicate with each other using the same SOAP interface.

In essence, SOAP defines how web services speak to each other by specifying a structured way for them to exchange information.

How does a SOAP API work?

SOAP-based APIs send and receive "envelopes" that contain well-formed XML. Developers create a request with the operation included in an envelope and in turn receive an envelope back following a requested operation. SOAP focuses on accessing named operations where each implements some logic through different interfaces specifying exactly how to encode a header using HTTP and XML.



SOAP messages follow a defined format using XML and always contain a SOAP envelope. The SOAP envelope contains an optional header element and a body that can contain any data that needs to be passed to the web service, for example, the developer API key or a phone number.

Structure of a SOAP message

SOAP-based APIs use remote procedure calls (RPCs) for client-server interactions. An RPC to a SOAP-based API goes, in general, like this:

- A client (your app or service) generates a request that creates a message in properly formatted XML, containing a SOAP envelope with a request for a service (such as fetching a piece of information based on an included piece of data, like the price of a book based on its ISBN number).
- This SOAP message gets sent to the server, where it's received and then sent to the appropriate server-side application that provides the requested service.

- The response from the service is returned to the client in an envelope, using properly formatted XML.

RESTful vs. SOAP APIs

RESTful APIs have become the prevalent style of APIs thanks to their popularity among developers, and some companies (such as Twitter, Google™ and Yahoo) now only offer RESTful APIs. Many developers prefer RESTful APIs because:

- In general, they are more intuitive to use and have a simpler set of commands for operations than SOAP APIs. This simplicity and ease of use (especially over many SOAP APIs) is the biggest reason for their popularity.
- They return a simpler response, using less bandwidth and reducing complexity for the developer.
- They can support different data formats, most typically, XML or JSON (see below).

JSON and XML—what are they?

JSON and XML are common standards used for data exchange with APIs. Since the SOAP protocol prescribes XML, the XML standard is always used for data transfer in SOAP-based APIs. For RESTful APIs, there is the flexibility to choose any data standard, but XML and JSON are the most common.

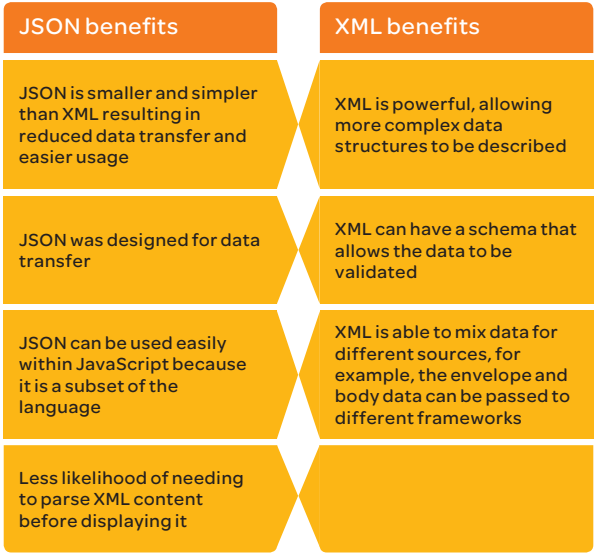
JSON is a text-based standard derived from JavaScript, and allows data to be represented using

objects. It was designed specifically for the purpose of enabling data transfer between servers and web applications.

XML is a markup language that is used in a variety of different technologies, including RSS and web-based APIs. XML was designed to represent data rather than display data like HTML and uses tags to define the structure of data.

JSON versus XML in APIs

There is a general trend towards usage of JSON over XML, mainly driven by developer preference. JSON has the main advantage that it integrates easily with JavaScript, a language that many developers are comfortable using. The overall benefits of the data standards are outlined in the table below.



API error handling

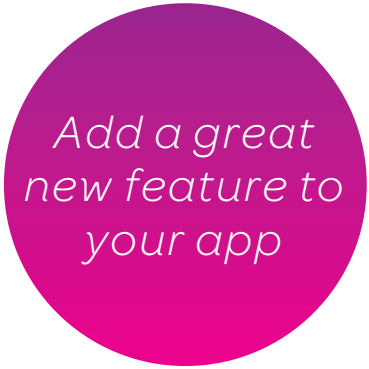
Error handling for RESTful APIs can differ and depends on the implementation of the API. The error response returned will usually include a standard HTTP Status Code and a message that expands on the status code. The message may simply contain some text or may include a web link directing the developer to further information about the error. HTTP Status Codes are defined by the IEFT, which works closely

with W3C and other web standards organizations, and have defined meanings.

For instance, a call to a RESTful API might return the error code 401, with the standardized meaning of "Unauthorized." This lets the developer know that they aren't authorized to perform the request they're making, and the provider could then point to additional information in its documentation, explaining the authentication requirements for its API.

When using SOAP-based APIs, the response returned by the API contains an envelope with the error information. The envelope contains a standard SOAP "Fault" structure that defines the error that has occurred. The "Fault" structure contains an error code and may contain text information with further details on the error.


Chapter 4
OTA
How To Use
External APIs in Your App or Service



With companies offering all sorts of services for mobile developers, using an external API (including network APIs) can help to simplify your development process or add a great new feature to your app. Rather than implementing a feature or service on your own within your app or on your own web back-end, many content and service providers offer easy to use APIs that can save you time and money. Building your own backend service or database can be complex. In this section, we will look at some of the different options for developers who need to use web-based services.

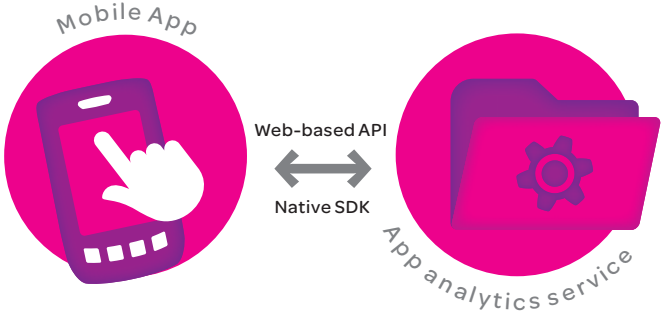
Architecting your apps and services to use external APIs

Integrating an API with your app will depend on the different options that the API provider gives you,

what you want your app to do, and how you integrate the APIs with your app. Below we examine some of the different ways that you can architect your app or service to utilize external APIs.

Mobile apps using external APIs

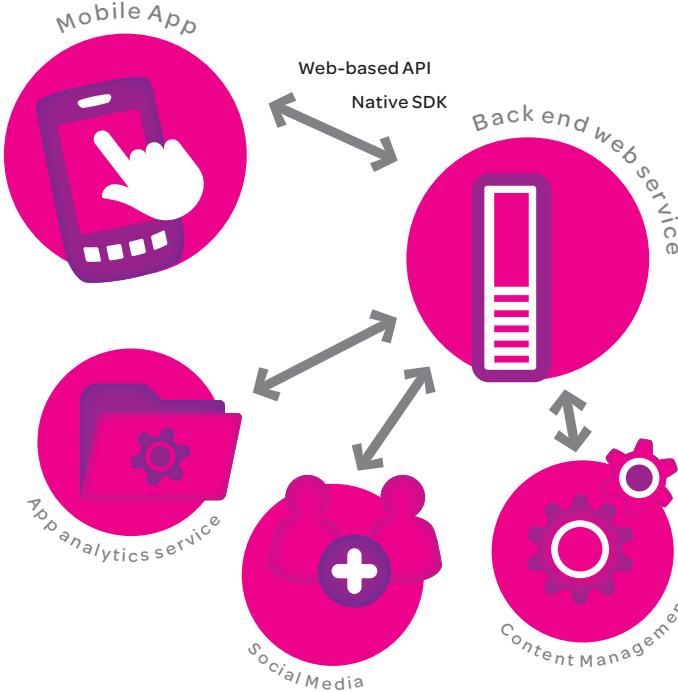
Applications can call web service APIs directly using either the device SDK (software development kit) or via the web-based API. All communication is between the web service and the app, and the app will store data relevant to the app or user on the device.



Mobile apps using a backend service and external APIs

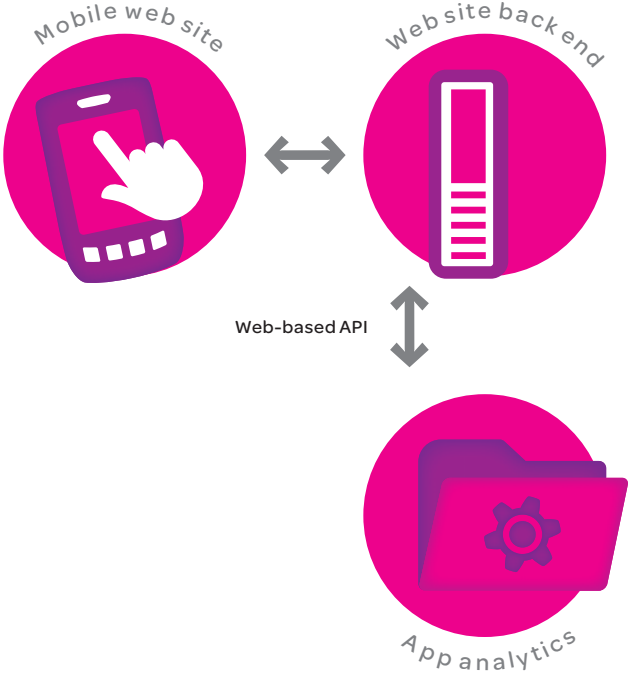
Developers can make their own backend service or use a PaaS (Platform as a Service) provider to direct calls through and manage authentication. The back-

end can store data about app users as well as content for the app, and then serve this data through APIs to client devices. For example, a news app can update the main view content on the web service, and the app will update this without requiring a new version installation.



Mobile web sites using external APIs

Any external APIs used within a mobile web site will be called from the backend server and content is served to the frontend. This is the same architecture as a non-mobile web site where the backend server deals with third-party services and content.



Using native SDKs

Some API providers, particularly those specifically targeting mobile developers, offer native SDK plug-ins for mobile developers, and this is most likely the easiest way to integrate an external API into your mobile app. Using a native SDK plug-in is as simple as including the libraries and calling the API as you would any other API you call using the device SDK. Just import the plug-in to your existing development environment, and you can start using the APIs in your application.

Using web-based APIs

API providers will offer a web-based RESTful and/or SOAP API, sometimes in addition to a native SDK plug-in. Integrating this type of API normally involves more work than using a native SDK plug-in, but it depends on the complexity of the API and what you need to do with it. There are plenty of simple web-based APIs that can be easily integrated into your mobile app using HTTP. Web-based APIs can be exercised using packaged HTTP requests and responses, and depending on the environment used, device platform SDKs can provide classes for handling HTTP functionalities.

Authentication and authorization

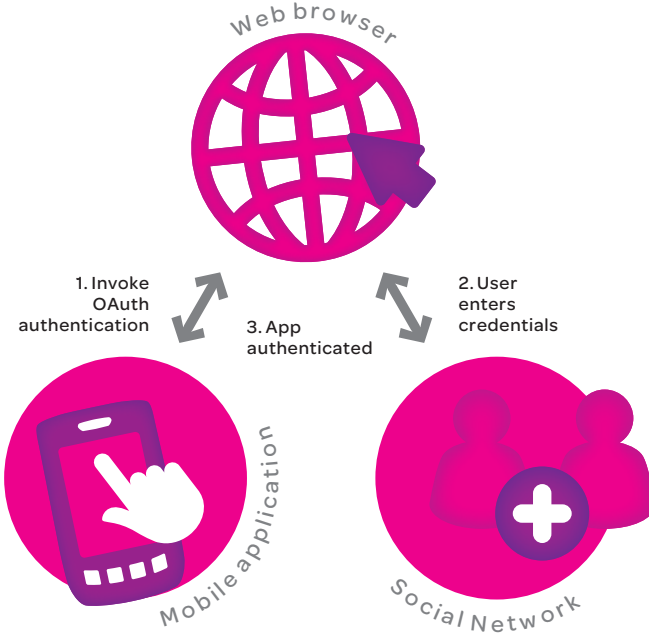
When using a web-based API or service, developers will be required to obtain an API key from the provider.

This key is used to authenticate a developer's application, and is the equivalent to a secret password when the app initiates a session with the API.

Another type of authentication is between the consumer that uses your app and an API provider. For example, if your application wants to get a user to post on their social network timeline, the user will need to log in and authenticate with that particular social network. The majority of web-based APIs use an open standard called OAuth for this authentication since this method never divulges the user's password to the application.

OAuth works by having a third-party service give an app an access token to authenticate a user, rather than divulging the user's password for that third-party service to the app. The user authorizes the app through the web browser or sometimes through the native app of the provider (e.g. Facebook). Your app will invoke this authentication process and deal with obtaining the access token that it will use to access the external API on behalf of the user. OAuth tokens can be revoked by the user at any time, and this is managed by the external API.

The diagram below shows the authentication steps using OAuth.



If you are using a native SDK for an external API, then the authentication is likely to be built into the API and the user will authenticate through a web view or through the web browser. However, if you are using a web-based API then you may have to implement your own way to deal with the authentication through a web view or browser.

Security

Apart from authentication with API providers, you should also ensure that your user data is protected and encrypted when appropriate. Although it is not always necessary to encrypt data using SSL and this method of transport is slower than standard HTTP calls, it is essential that you encrypt any sensitive data such as passwords or other confidential or personal information.

Tools and IDEs

If you are using an API that provides a native SDK, you can continue to use the same tools and IDE as you already use for your app development. If you are using an external web-based API, you can use the built-in HTTP classes and objects to construct the requests and parse responses. Some platforms even provide a class to encapsulate JSON response data to make it easier to manipulate.

Many external APIs support both XML and JSON and will also provide sample code in languages such as Ruby, PHP and Python. Some will even offer an in-browser console to try out the API with different requests without doing any coding. This is a useful way to try out the API, but is usually restricted to simple operations such as fetching data.


Chapter 5
May
AT&T
Network APIs

AT&T gives developers easy access to a number of different APIs in a single place, and provides SDKs, development tools and other resources to make working with the APIs fast and simple. The APIs provide a number of network-based services that can uniquely be provided by AT&T for customers on its network, including some location services, messaging and others. In addition, there are APIs that can be accessed by apps regardless of the user's network, such as the Speech API.

The range of APIs offered by AT&T is always growing. So be sure to check with the AT&T Developer Program site to see what's available in Beta and Production, and with the AT&T Foundry program to see what's being tested in Alpha. These are the AT&T APIs currently available on the AT&T API Platform:

- Speech
- In- App Messaging From Mobile Number
- Device Capabilities
- Call Management

- Location
- Payment
- SMS
- MMS
- U-verse® Enabled.

You can learn about the capabilities of each one of the APIs below, as well as the available formats and SDK support. In Chapter 6 we go through a number of exciting options of what you can do with these APIs!

Speech

Voice control and interaction is becoming more and more popular among mobile app users, and the AT&T Speech API lets you build this feature into your apps. It's pretty simple: your app sends audio to the API, and the API returns text of what your user said. The API is powered by the AT&T Watson℠ speech engine and supports several different contexts, which help the engine come up with the most accurate transcription. The AT&T Speech API is available across networks.

In-App Messaging From Mobile Number

This API allows your app to send SMS and MMS messages from within the application, and they will appear to the recipient as though they were sent from the original user's AT&T mobile phone number. This means you can share content from within your app, or even send notifications, in a widely supported

format, without your users ever having to leave the app itself (a simpler user experience). You will need to get users' consent to send and receive messages on their behalf, but AT&T will obtain and maintain this consent information on your behalf.

"Share via SMS" is a common feature in apps, but typically it copies some text to the device's SMS messaging application, where the user must then send it. Rather than requiring that extra step, the AT&T In-App Messaging from Mobile Number API allows your app to send content without ever leaving the app thus making it more user friendly, and increasing the usage of the app.

Currently, messages can be sent to any North American (US and Canada) number, or an email address or short code.

Device Capabilities

The Device Capabilities API does what it says in the name: it allows you to retrieve the make, model, firmware version, GPS/A-GPS functionality, web browser and MMS capability of a device on the AT&T network. This means that you can:

- Optimize the content you deliver to a user, based on his or her device's capabilities
- Utilize the correct APIs based on the device (for instance, network location vs. built-in GPS)
- Better understand your user base and their devices

to help inform your development choices
- Target your application to particular device configurations.

Call Management

The Call Management API lets you easily add voice call and SMS support to your apps. You can add interactive voice response, speech recognition, voice-to-text, SMS, and more, and it lets you create outgoing call sessions and receive incoming calls from virtual numbers as well. This means you can build a conference call system, a virtual PBX and voicemail system, IVR apps that let users navigate via voice commands, group messaging and more.

The API uses simple commands, such as "say," "answer" and "transfer," making it easy to add the functionality to your apps and services.

Location

The Location API gives you location information on AT&T subscribers (who have given you consent), but unlike using a device's GPS, it can be used by web-based services (think of all the back-end processing you can do), and works on devices without GPS. No client or agent

needs to be present on the device and no device power is consumed. Plus, this API can work without an application open and running on the device, allowing you to create novel and powerful location-based services that can operate automatically for the end users.

You can also choose from three different levels of accuracy depending on your needs and the device's status (for example, if the device is indoors, the most accurate level may not be available):

- **City** (generally 10,000–20,000m / 3–6 miles)
- **Neighborhood** (generally 500–10,000m / 1/4–3 miles)
- **Intersection** (generally 25–500m / 50–165 feet).

Another bonus: You will need to get users' permission to track their device's location, but AT&T will manage the consent part of the process for you, meaning you don't have to track it yourself

Payment

The Payment API allows you to add in-app purchases, with a convenient and secure billing option to your apps, letting you use what is currently the most popular app business model, according to data from the analytics company Flurry[1], and others. Using the

1) http://blog.flurry.com/bid/65656/Free-to-play-Revenue-Overtakes-Premium-Revenue-in-the-App-Store

"Freemium" model, in which your app is free but premium features or content (like new game levels or feature upgrades) are sold within the app, has proven to be very successful for many game and app developers.

What makes the AT&T Payment API stand out over some other payments systems is that it lets users charge purchases directly to their AT&T mobile bill (this is called carrier or operator billing). Carrier billing has been shown to increase conversions 1-3x over credit-card billing and other methods, according to BlackBerry[1]. This is because it's got a lot less "friction"—meaning it's just way easier for customers to use. With carrier billing, users don't have to type in a credit card number or other information; the API knows who they are, and bills their account accordingly.

With the Payment API, you can sell digital and virtual goods and content in your app, and it also supports subscriptions. If you have a mobile website, you can easily integrate carrier billing and avoid application stores entirely. You'll also receive a revenue share of 70%, and won't incur any setup or ongoing charges for using it.

1) http://devblog.blackberry.com/2012/11/app-world-integrated-carrier-billing/

SMS

The SMS API lets your app send SMS messages to and receive messages from AT&T subscribers, including group messages. It also works with short codes, and supports delivery receipts, so you know that your message got through. This API also automatically gives developers short codes to use at no additional cost, and allows them to choose a Free to End User model so their users won't be charged for messages sent or received.

MMS

The MMS API lets you send and receive multimedia messages to and from AT&T subscribers. With it, you can send and receive photos, videos and audio, even from feature phones (as long as they support MMS). This API also lets you check network delivery status so you can ensure your messages were received.

U-verse® Enabled

For your users who have AT&T U-verse TV service, the AT&T U-verse Enabled APIs give you the ability to connect your apps to their receiver, display content on their television, and interact with their TV box. When users are on their home Wi-Fi network along with their U-verse receiver, your apps can send remote control commands, play music and videos and show photos on the their TV, detect channel changes and get program guide data.

AT&T mHealth Platform

AT&T has been working with app developers to create brand new applications that utilize health and wellness data. Applications that are mHealth-enabled are given access to data (with the user's permission) that exists about the user, enabling people to interact with their health data in applications that understand them!

To sign up for the mHealth development platform, visit http://mhealth.att.com/dev.

SDKs and Tools

To make things even easier for you, AT&T provides a number of SDKs, tools and plug-ins for its APIs, supporting several platforms and development tools. One of the biggest advantages for using these tools is helping you more efficiently implement their solution, saving time and cost as well.

The SDKs include:

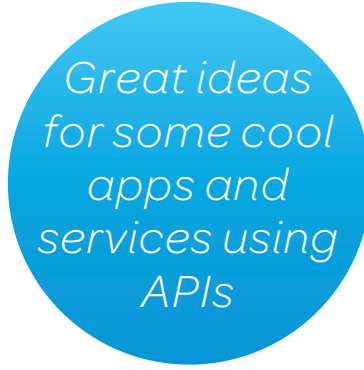
- HTML5 SDK
- Microsoft® SDK
- Android SDK
- iOS SDK.

The tools and plug-ins include:

- A toolkit for the Salesforce platform
- An adapter for IBM® Worklight®
- A module for Appcelerator® Titanium

- An SDK for Microsoft Windows that includes Visual Studio® extensions
- A plug-in for Adobe® PhoneGap®
- A plug-in for Tiggzi
- A plug-in from Viafo-
- A Sencha Touch SDK.


Chapter 6
Web Work Order
Use
Cases and Examples



Hopefully by this point you're starting to come up with some great ideas of your own for some cool apps and services you can create using APIs. In this chapter, we'll provide some specific use cases and examples for how the AT&T APIs can be used or mashed up to create some creative services that will wow your users! First, we'll break out each AT&T API and illustrate some potential ways they can be put to use. We'll then introduce some mashups—cool app ideas that combine multiple APIs!

AT&T API Potential Uses

Speech

Use	Market
A voicemail-to-email service	Consumer Enterprise
Voice UI for kids, seniors or those with disabilities	Consumer Health Care
Control your TV remote or other household gadgets with your voice!	Consumer
A messaging application that reads incoming text and creates outgoing text from audio	Consumer Enterprise

In-App Messaging from Mobile Number

Use	Market
Messaging applications with a customized UI	Consumer Brand
Voting apps that integrate with existing SMS voting systems	Consumer Enterprise Entertainment
Social networking apps that share content via SMS or MMS	Consumer Education
Send web shortcuts and links to phones via SMS	Enterprise
Share photos and videos from within apps via MMS for photo blogging	Consumer Healthcare Education Field Worker
Smart group messaging apps	Education Entertainment Field Worker

Device Capabilities

Use	Market
A video application that can send different formats of video to different types of devices	Consumer
Determine optimized location function (onboard GPS, or network location APIs)	Consumer Games
Analytics to know your users and have a feedback loop for your app lifecycle	Consumer Enterprise

Call Management

Use	Market
On-call app that routes incoming support calls to the right contact based on the time of day or expertise needed	Enterprise Field Management Health Care
Creating a voicemail system with message transcription	Consumer Enterprise Health care
Building a party notification app—conference call/ group messaging application	Consumer
Make voice translation apps	Consumer Enterprise Education

Location

Use	Market
Asset tracking the location of M2M devices	Enterprise
Remote pet tracking	Consumer
Geo-fencing messages—notifications when near a specific area or get near a certain point	Consumer Enterprise
Location-based multiplayer games	Consumer Games
Location-based advertising	Business to Consumer

U-Verse® Enabled

Use	Market
Make TV social—determining what a user is watching, then let them share the information with their social networks	Enterprise
Multi player games on a TV screen	Consumer Games
Photo/video slideshow	Consumer Enterprise
Customized remote control / parental controls	Consumer Games

Payment

Use	Market
Sell new levels in a game, or power-ups	Consumer Gaming
Sell music or videos within your app	Consumer
Charge for a subscription-based news service	Enterprise
Sell access to new features such as photo filters, personalization items and more	Enterprise
Sell access to analytics or more storage	Enterprise

SMS

Use	Market
Add security and identify verification	Consumer Enterprise
Send news alerts or score updates	Consumer Enterprise
Conduct voting and polling	Entertainment Politics Business to Consumer
Share music, photos and more	Consumer Enterprise Education
Send message or alert from a web service and confirm receipt	Consumer Enterprise Health care

MMS

Share content from a web service to users, even on feature phones	Enterprise
Receive pictures and video to project on a large screen at an event	Consumer
Receive pictures or video for remote teaching	Education
Send picture of video for medical support	Health care

Example Apps and Mashups With the AT&T APIs

Speech	In-App Messaging	Device Capabilities	Call Management
Voice-enabled Remote Control	Advanced Fleet Management	Turn-by-Turn Navigation for Any Device	Package Delivery
Smartphone Helper	Meetup Mashup	Smartphone Helper	Shopping Assistant
	Social Sports Watcher	Social Sports Watcher	Advanced Fleet Management
			Turn-by-Turn Navigation for Any Device
			Small Business Marketing and Communications Service
			Smartphone Helper

The colored dots within the boxes correspond to the color designated for the AT&T API, indicating their use in the Apps and Mashups. Many apps use two or more than APIs!

Location	U-Verse Enabled	Payment	SMS/MMS
Meetup Mashup	Voice-enabled Remote Control	Turn-by-Turn Navigation for Any Device	We the People
Package Delivery	Party Screen	Wi-Fi Hotspot Login	Turn-by-Turn Navigation for Any Device
Turn-by-Turn Navigation for Any Device	Social Sports Watcher		Package Delivery
We the People			Wi-Fi Hotspot Login
Shopping Assistant			Small Business Marketing and Communi-cations Service
Advanced Fleet Management			Party Screen

Mashups Using the AT&T APIs

APIs get even more exciting when you combine them to create mashup apps. We've listed several such mashup ideas below, and summarized them in the table on the previous pages. Each column represents an API, and they're color coded (corresponding to the tables in the previous section) to help illustrate which APIs are being mashed up. For instance, under the Speech API column, "Voice-enabled Remote Control" is listed, and the yellow colored dot corresponds to the U-Verse Enabled API, since that's the other API that's mashed up to create the app.

Voice-enabled U-verse® Remote Control

APIs used: Speech and U-verse® Enabled

Voice control is a hot topic for mobile developers, especially when it's coupled with an intelligent service that can understand users' requests, rather than just navigate them through a menu. What's even better is when voice control can replace a painful and slow text entry—such as when you're trying to search for a show on your TV's electronic program guide and entering titles or keywords on its remote control!

This app would run on a user's mobile device, and would utilize the AT&T Speech and U-Verse Enabled APIs to let users search for shows, control their U-verse set-top box and more by speaking into their device. So instead of having to navigate to the search function on their TV remote, then triple-tap in "cooking

shows," they would fire up this app, and speak the search term into it. The app would send the recording to the Speech API, which would transcribe it, using its "TV" context to ensure an accurate translation.

The app takes the transcribed speech returned by the Speech API and searches guide data to return matches. Users can then choose to tune to the program immediately, or set a recording on their DVR. So a user could simply speak into their device "Mario Batali cooking show," the Speech API would then transcribe that speech on the fly, send the text back to the app, which then searches guide data and returns the shows meeting the criteria in the text—which is much simpler than trying to type in the search!

Meetup Mashup

APIs used: Location and In-App Messaging from Mobile Number

Location-based apps are essential for anyone on the go, or in other words, anyone with a mobile device. For highly social users, being in touch with friends and family easily is the key benefit for having a smartphone. Think about the benefits of connecting those social users in impromptu scenarios—you're about town and discover a cool event you are excited to attend and it's happening in an hour! What if you could click on an app and see whether anyone in your social circle is in the area and send a text to them all inviting them to meet you at a location.

This app would utilize the AT&T Location API and would alert the user which of their friends are in the area. Via the In-App Messaging from Mobile Number API, the user could easily send a group message to all his friends in the area with an invite to the location for the meetup. Using this API also provides them with an easy way to respond with "Yes, I'll be there" or "Sorry, can't make it" responses.

The benefit of the In-App Messaging from Mobile Number API is that it takes the "Share Via SMS" feature and rather than implementing another step for the user to copy and send the message, it would automatically send the message from within the app itself. Integrating this app with popular social APIs could make this app sticky, usable and a must-have for the social butterflies out there. Rather than just checking in, you can invite your friends in the area easily, all without ever leaving your application.

We the People

APIs used: SMS API, Location API

One clever use of APIs could help power an application that connects users with their local politicians. If a local politician wanted to connect more easily with their constituents, they could use a variety of APIs to make them a shoo-in* in the next election.

(*Results may vary :-)

Politicians could use the SMS API to send out voting polls to constituents on important matters, and could easily get their feedback and be better informed on the issues most important to voters.

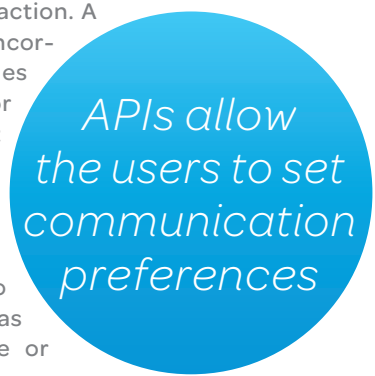
The app could be a hyper-local informative app that could use the Location API to help identify core voting areas and alert them to Town Hall meetings that they can attend. Additionally, you could provide a Mapping Service API to ensure that they get there.

Package Delivery

APIs used: Call Management API, SMS API and Location API

Online retailers may want to incorporate a unique approach to attract or keep loyal customers. Providing an app that helps users manage and track package delivery and package returns could be an ideal way to ensure customer satisfaction. A developer may want to incorporate a system that enables users to set preferences for delivery features that fit into their life.

The application could use several APIs (Call Management or SMS APIs) that would allow for the user to set up preferences, such as receiving a text message or



voice message that their package has been shipped, is available for tracking, and when it will arrive. For sensitive or large deliveries, it could provide a way to schedule deliveries, re-route them or authorize release for convenience to the user.

Most users have preferences for how and when they like their shipments received and might prefer to set a preference to receive them at a specific date and time, after allowing for general delivery times. Additionally, it could easily alert users when shipment is delayed due to inclement weather.

Shopping Assistant

APIs used: Call Management, Location

As a brick and mortar retailer, building value in an app rather than just having an app can be helpful in attracting loyal users. What if you could add functionality to an app that would aid in customer service and serve as a shopping assistant? Not only would your app show location of stores, but you could add features like checking inventory at various locations, schedule and track shipments, and use in-app management to obtain feedback on recent visits, purchases, and etc., that would reward valuable customers.

With the AT&T Call Management API, you could provide easy access to many customer service functions directly within the app that could prompt a user to call a store and check on inventory before they

make the trip. In addition, the app could identify when a user entered the store via the Location API and prompt the user within the app about store specials, or gather feedback about their visit with no friction for the user.

Advanced Fleet Management

APIs used: Call Management, Location, In-App Messaging from Mobile Number

This app/service would be used by companies with many drivers or vehicles out on the road. Each driver's device would be used to track a vehicle's location, via the Location API, allowing the fleet manager to know where the vehicles are located at a given time. By integrating with the Call Management API, the manager can easily reach a specific vehicle/driver through a unified interface, rather than having to locate the vehicle, determine the driver, look up their number and manually call.

Communications could be done via voice or text, and routes could be updated in real-time. In addition, if the drivers were performing delivery services, they could send out notifications from their device to the mobile devices of shipments' senders or recipients, letting them know a delivery had been made.

Turn-by-Turn Navigation for Any Device

APIs used: Device Capabilities, Location, Call Management, SMS, Payment

While turn-by-turn navigation with audible spoken directions is a common feature in smartphone apps, it is less common on feature phones. This service would use the Device Capabilities API to determine if a user's device had onboard GPS. If so, that would be used to determine the user's location; if not, it would utilize the ability of the AT&T Location API to determine the location of such devices, and could use the Call Management API to provide a voice-driven interface for users to search for a destination and receive directions.

Additionally, directions could be texted via the SMS API, and the developer could charge for the service on a per use or subscription basis with the AT&T Payment API.

Wi-Fi Hotspot Login

APIs used: Payment, SMS

In this scenario, a user would try to join a paid Wi-Fi hotspot with their mobile device; the landing screen would indicate the charges, and for AT&T users, give them the option to bill the purchase to their AT&T Wireless account. When using this option, the user would authenticate and accept payment, then receive an access code via SMS, which they would then use to log in to the hotspot. The registration and payment is

far easier via this method than using credit card billing, offering an improved experience for users and a potentially higher conversion rate for the hotspot provider.

Small Business Marketing and Communications Service

APIs used: SMS, MMS, Call Management

With the suite of AT&T APIs, it's possible for a developer to create an integrated marketing and communications system for small businesses. Such a service would support:

- Inbound and outbound SMS marketing campaigns
- Social media integration, allowing customers to submit photos and videos via MMS for display in the business and on its social media sites
- A virtual PBX and voicemail system, with IVR menus and other services.

Smartphone Helper

APIs used: Device Capabilities, Speech, Call Management

For many users, learning to use their first (or simply just a new) smartphone can be difficult. The Smartphone Helper app would provide users with voice-based help services, both allowing them to navigate by voice (such as by answering the question, "What do you want to do?" or "What do you need help doing?"), and receive voice coaching on how to

complete a task. The Device Capabilities API would be used to determine the device a user has. By receiving voice coaching, the user could listen to the instructions while navigating a task, rather than having to switch back and forth between instructions and the task itself, as they would with a text-based solution.

Party Screen

APIs used: U-verse® Enabled, MMS

This API could be used by U-verse subscribers to run a party photo/video screen at their home or business. For instance, a person could have a "virtual photobooth" at their party, and guests could submit their photos via MMS. The images could be forwarded to an app on the host's mobile device, where the host could moderate or publish them, and then display them on a TV connected to a U-verse set-top box. In addition, backdrops and other elements could be added by the service. For instance, if it was an Oscars® party, a background made to look like the red-carpet entrance could be added.

Social Sports Watcher

APIs used: U-verse® Enabled, Device Capabilities, In-App Messaging

Watching sports is more fun with friends. With this app, you can share the experience with your friends, even if you're not in the same place. For users with U-verse TV service and a compatible device, the app would detect what they were watching, and offer them the option to start a social watching session with certain friends of their choice, who would receive a message saying their friend wanted to watch a football game or another event along with them.

The Device Capabilities API could be used to detect if the user had a device compatible with the app; if so, the user would get a message asking if they wanted to download the app, which they could use to communicate. If not, they could still participate via SMS. Users could trade jabs about their favorite teams, talk trash about star players and share in the excitement of watching their team win. The app could also pull in additional content from social networks and sports news sites.

What's Next?

AT&T provides a set of unique capabilities for you to take advantage of through APIs like Speech, Device Capabilities, Location, and U-verse Enabled—capabilities you won't find anywhere else. These APIs give you the ability to offer new features and functionality for your users!

Other APIs, like Call Management, Payment, and In-App Messaging from Mobile Number give you ways to deliver the best user experience possible to the millions of subscribers on the AT&T network. Optimizing your apps for AT&T subscribers can help create deeper engagement, more loyal users and higher revenues.

AT&T is committed to delivering SDKs, plug-ins and other resources to make your development around the APIs as smooth as possible. All the APIs are available through a single place—the AT&T Developer Program—and there's a free trial available. We can't wait to see the creative and exciting applications you develop with the AT&T APIs, so jump in and start coding!