



Using the Google Maps API to book the perfect ride.

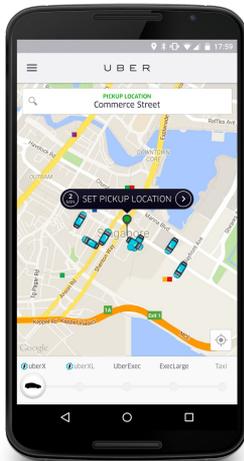
Target customers and optimise vehicles using the Google Maps APIs.

The number of transportation booking apps—taxi, rideshare, shuttle and limo to name a few, has exploded over the last few years primarily due to the increase in GPS enabled smartphones, fast and reliable mobile connectivity and simple, secure payment options. These apps offer real benefits to customers, drivers and taxi booking companies and location plays a critical role for all of them. Google Maps APIs enable many feature-rich capabilities that can provide the best location-based information and user experiences for booking and tracking the perfect ride.



This paper will describe how the different **Google Maps API** can provide the best location-based information and user experiences for booking and tracking the perfect ride, leading to greater customer satisfaction, optimised routing and ride matching as well as better driving analytics.

Use the Maps API to show the real-time location of the customer and driver.



The Uber app shows all available taxis in the customer's vicinity.

Help reassure the customer that their ride is on the way by showing real-time location of the vehicle on a map—in relation to their position. If the driver is unfamiliar with the area this can also help guide her to the customer's location. Google has global coverage so if you're working in multiple countries we've got you covered. No need to work with multiple map providers.

Asset tracking with Google Maps allows a taxi booking company to view the real-time location and status of all their taxis on a central map to help process offline bookings coming in via telephone calls. Need help storing and retrieving all of your taxi fleet locations? Take a look at our [blog post](#) on how to store your location data in the cloud with Firebase.



Add a map into any application.

Google Maps APIs are available for Android, iOS, web browsers and via HTTP. All of the services allow you to embed maps within your application on each platform and place markers on top of the maps showing the location of any customers or vehicles.

You can customise your markers by changing the default color (possibly showing the time a person has been waiting) or replacing the marker with a custom image (such as the type of vehicle used by your driver).

Info windows, or popup windows, can provide additional information to the customer, such as the type of car, registration, driver rating or photograph.

You can find more details about adding maps into your application within the help documentation: <https://developers.google.com/maps/>

Use the Places API for fast booking and easy address entry.

The type-ahead-search suggestions available with Places API Autocomplete makes it fast and easy for the customer to provide an accurate pick-up and drop-off location with minimal keystrokes. The Autocomplete feature is available in [Android](#), [iOS](#), [Javascript](#) and as a [web service](#) for flexible integration options.

If booking a taxi immediately, allow your customer to automatically communicate their current location using the location capabilities of their smartphone or browser. On Android and iOS make use of the [Places API Place Picker UI](#) widget so your customer can choose their location from a list of nearby places. Alternatively, the [Geocoding API](#) can provide the nearest address to their current location.



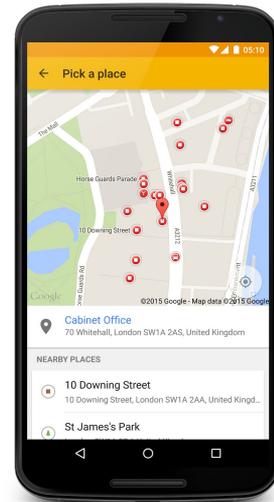
Enrich your application with places.

The Places API is available on both Android and iOS to allow your users to get access to the rich set of data that powers Google Maps. Places features over 100 million business and points of interest and are constantly updated.

For autocomplete you can use a UI control to provide a simple, and quick to implement, autocomplete feature within your application. For more complicated and controllable functionality and user experience you may programmatically control the information you receive from the service.

You can get more information about the Places Autocomplete and Picker functionality at the following locations.

- [Google Places API for Android](#)
- [Google Places API for iOS](#)



The Android PlacePicker UI widget.

Use the Distance Matrix API to find the closest taxis based on drive time and drive distance.



Performing a simple radius or “straight line” distance search to find the nearest available taxis can often provide misleading results. For example, a taxi may be on the far side of a highway or the other side of a river and take a long time to drive to the pick-up point. The [Distance Matrix API](#) can calculate the driving distances and times, based on Google’s road network, to improve the accuracy of your search results. The distance matrix API can also make use of the real time traffic information provided by Google which can improve the matches between drivers and customers.

Also if you have multiple vehicle types available, you could use the Distance Matrix API to show the closest of each of these types so that the user could make an informed decision not only on time they would take to come but also on the type of vehicle to hire also.



Identify the nearest vehicle with the distance matrix.

The distance matrix API works by taking a list of origin locations and a list of destination locations, either of which could be used for the customer or vehicles depending on the viewer of the application (the origin might be the vehicle looking for customers or the customer looking for vehicles).

Origins and destinations can be specified as coordinate pairs or as addresses, the latter will be matched when the service is called.

As with the Google Directions API the Distance Matrix API can specify parameters to generate results based upon current traffic from the location of the vehicle.

The following URL shows the call between a person standing at Euston Station and a number of vehicles on surrounding roads.

<https://maps.googleapis.com/maps/api/distancematrix/json?units=imperial&origins=51.527699,-0.132660&destinations=51.525950,-0.136454|51.525571,-0.128854|51.529131,-0.131603>

Responses from the Distance Matrix API consist of a simple match between the origin and destination and the time resolved.

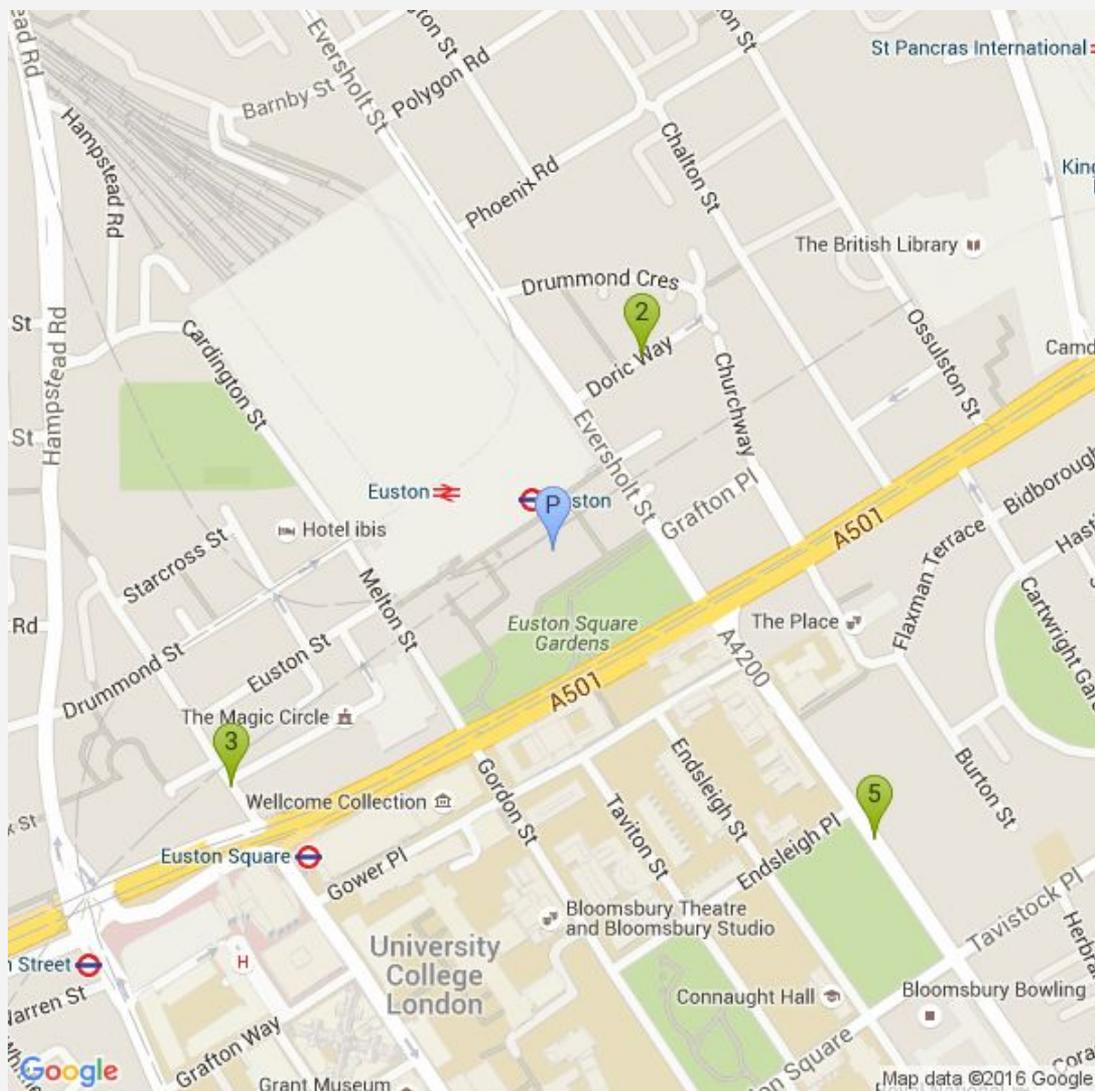
```
{
  "destination_addresses" : [
    "152 N Gower St, Kings Cross, London NW1, UK",
    "20 Tavistock Square, Kings Cross, London WC1H 9HW, UK",
    "79 Doric Way, Kings Cross, London NW1, UK"
  ],
  "origin_addresses" : [ "1 Eversholt St, Kings Cross, London NW1 2DN, UK" ],
  "rows" : [
    {
      "elements" : [
        {
          "distance" : {
            "text" : "0.3 mi",
            "value" : 510
          },
          "duration" : {
            "text" : "2 mins",
            "value" : 141
          },
          "status" : "OK"
        },
        {
          "distance" : {
            "text" : "0.4 mi",
            "value" : 705
          },
          "duration" : {
            "text" : "3 mins",
            "value" : 180
          },
          "status" : "OK"
        }
      ]
    }
  ]
}
```

```

    "distance" : {
      "text" : "0.6 mi",
      "value" : 923
    },
    "duration" : {
      "text" : "5 mins",
      "value" : 285
    },
    "status" : "OK"
  }
]
},
"status" : "OK"
}

```

This could be displayed as follows, showing locations and potential times:



On the web the Distance Matrix API can be used with the DistanceMatrix JavaScript service, you can find out more at the location below:

https://developers.google.com/maps/documentation/javascript/distancematrix#distance_matrix

On Android you can use the Client Libraries to easily connect to the web services, this is especially useful if you are using the Google Maps API on the Premium Plan where a proxy service should be used to contain the cryptographic key that is used when authenticating access to the web service and should not be shared within a mobile application directly. You can find out more at the following resource:

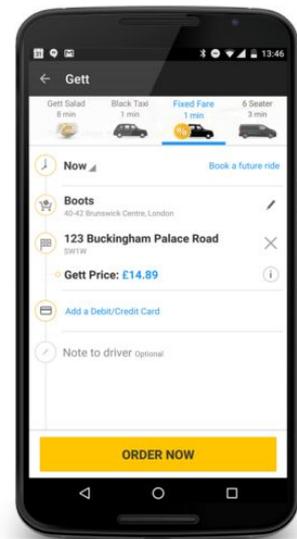
<https://developers.google.com/maps/web-services/client-library#requirements>

Use the Directions API to provide arrival times and fare calculations to reassure the customer.

If your customer is in a hurry, it can feel really frustrating wondering when her taxi will arrive. Keep the customer informed of the vehicle's estimated time of arrival using the duration-in-traffic information in the [Directions API](#). This can be done at the moment a person requests a vehicle using data about current road conditions or could be done ahead of time (for future bookings) using data about historic conditions.

Even before the journey starts, customers can be provided with an estimated fare based on the optimal route and traffic conditions and a visual representation of the route. Alternative routes can also be provided. These details can be used to estimate when the customer may arrive at their destination and how much it should cost.

Whilst the customer is travelling this information can be constantly updated if required showing the progress towards the destination and also any potential upcoming issues. Allowing the customer to be kept informed before and during the ride will increase confidence and improve the customer's user experience.



Add the Directions API to your application

The directions API can be added to your application in the same way as the Distance Matrix API above and called in a similar manner:

<https://maps.googleapis.com/maps/api/directions/json?units=imperial&origin=51.527699,-0.132660&destination=51.494833,-0.146312>

You need to provide an origin and a destination point as well as a number of optional parameters which might include:

- waypoints — Specifies an array of waypoints. Waypoints alter a route by routing it through the specified location(s).
 - This can be useful if you want to drop off or pick up at a number of locations.
- alternatives — If set to true, specifies that the Directions service may provide more than one route alternative in the response.
 - This can be useful if you want to show a number of options for drop off to the user and allow them choice (or to show that the usual quicker route is not necessarily the best one).

- language – Specifies the language in which to return results.
 - This may be useful when you are providing details to people of different nationalities.
- departure_time – Specifies the desired time of departure.
 - This can be useful if the user is booking in the future and wants to see the time it might take at that exact time.
- traffic_model (defaults to best_guess) – Specifies the assumptions to use when calculating time in traffic.
 - You can be either optimistic or pessimistic on the time returned when looking at future events, or show the range to the user which might be useful in planning.

The response from the Directions API can be quite long, some of the most important elements that can be immediately shown to the user would be the distance and duration as given below:

```
"distance" : {
  "text" : "4.1 mi",
  "value" : 6562
},
"duration" : {
  "text" : "21 mins",
  "value" : 1250
},
"end_address" : "76-98 Buckingham Palace Rd, London SW1W, UK",
"end_location" : {
  "lat" : 51.4948133,
  "lng" : -0.1462403
},
"start_address" : "1 Eversholt St, Kings Cross, London NW1 2DN, UK",
"start_location" : {
  "lat" : 51.5278751,
  "lng" : -0.1328546
},
```

Full information about everything that is returned is given at the following location:

<https://developers.google.com/maps/documentation/directions/intro#Introduction>

First allow your customers to book the perfect ride, next power your back office to understand customers better.

We hope you're able to incorporate some of these ideas into your transportation applications. Using the Google Maps API for your customer facing applications may be the first start to incorporating intelligent location into your business. You can further enhance how your back office applications to better understand how your driver's perform on the road or tidy up reports of past trips for better accuracy or visualisation using functionality such as the [Roads API](#). The Google Maps API's can be used for all aspects of your business.