Continue

If working with Basic4Android and looking for an easy fix to the problem, try this it works both Google maps and Openstreet even though OSM creates a bit of a messy result and thanx to [yndolok] for the google marker GooglemLoc=" &[Latitude]&"+"&[Longitude]&"/@"&[Latitude]&","&[Longitude]&",15z" GooglemRute=" &[FrmLatt]&","&[FrmLong]&"/"&[ToLatt]&","&[FrmLong]&"/@"&[ScreenX]&","&[ScreenY]&",14z/data=!3m1!4b1!4m2!4m1!3e0?hl=en" 'route ?hl=en OpenStreetLoc=" 16/"&[Latitude]&"/"&[Longitude]&"&layers=N" OpenStreetRute=" It's easy to embed Google Maps in a few different ways. Shutterstock This story is available exclusively to Business Insider subscribers. Become an Insider and start reading now. Embedding a Google Map into a personal website or blog is a simple way to help visitors navigate an area or easily find directions to a location. Google Maps has a "Share or embed map" feature that provides users with the map's HTML code so that they can insert it into their website's code. You can embed a map view image, a street view image, or even specific directions for a route. Here's how to do it. How to embed a Google Map 1. Type Maps.Google.com into your web browser. 2. Go to the directions, map view, or street view image that you want to embed. 3. Click on the "Menu" tab in the top left corner. The icon is three horizontal lines. Click on the Menu icon. Kelly Laffey/Business Insider 4. Click on "Share or embed map." Find the option toward the bottom to share or embed. Kelly Laffey/Business Insider 5. The pop-up box will automatically highlight the "Send a link" tab, which enables users to email the map, or to share it on Facebook or Twitter. To get the map's HTML code, select the "Embed a Map" tab. 6. You can choose the size of the map (small, medium, large, or custom size) by clicking on the drop-down menu next to the HTML code. Go to "Embed a map" and copy the HTML code. Kelly Laffey/Business Insider 7. Copy the HTML code in the box, and paste it into the HTML section of your website. By embedding a map, you are agreeing to Google's terms of service. Note that live traffic information and other data may not be available in embedded maps. Related coverage from How To Do Everything: Tech: If working with Basic4Android and looking for an easy fix to the problem, try this it works both Google maps and Openstreet even though OSM creates a bit of a messy result and thanx to [yndolok] for the google marker GooglemLoc=" &[Latitude]&"+"&[Longitude]&"/@"&[Latitude]&","&[Longitude]&",15z" GooglemRute=" &[FrmLatt]&","&[FrmLong]&"/"&[ToLatt]&","&[FrmLong]&"/@"&[ScreenX]&","&[ScreenY]&",14z/data=!3m1!4b1!4m2!4m1!3e0?hl=en" 'route ?hl=en OpenStreetLoc=" 16/"&[Latitude]&"/"&[Longitude]&"&layers=N" OpenStreetRute=" The maps URL scheme is used to show geographical locations and to generate driving directions between two points. If your app includes address or location information, you can use map links to open that information in the Maps app in iOS or macOS. Unlike some schemes, map URLs do not start with a "maps" scheme identifier. Instead, map links are specified as regular http links and are opened either in Safari or the Maps app on the target platform. Table 5-1 lists the supported parameters along with a brief description of each. Table 5-1 Supported Apple Maps parametersParameterMeaningValuestThe map type. If you don't specify one of the documented values, the current map type is used.qThe query. This parameter is treated as if its value had been typed into the Maps search field by the user. Note that q=* is not supportedThe q parameter can also be used as a label if the location is explicitly defined in the ll or address parameters.A URL-encoded string that describes the search object, such as "pizza," or an address to be geocodedaddressThe address. Using the address parameter simply displays the specified location, it does not perform a search for the location.An address string that geolocation can understand.nearA hint used during search. If the sll parameter is missing or its value is incomplete, the value of near is used instead.A comma-separated pair of floating point values that represent latitude and longitude (in that order).llThe location around which the map should be centered.The ll parameter can also represent a pin location when you use the q parameter to specify a name.A comma-separated pair of floating point values that represent latitude and longitude (in that order).zThe zoom level. You can use the z parameter only when you also use the ll parameter; in particular, you can't use z in combination with the spn or sspn parameters.A floating point value between 2 and 21 that defines the area around the center point that should be displayed.spnThe area around the center point, or span. The center point is specified by the ll parameter.You can't use the spn parameter in combination with the z parameter.A coordinate span (see MKCoordinateSpan) denoting a latitudinal delta and a longitudinal delta.saddrThe source address to be used as the starting point for directions.A complete directions request includes the saddr, daddr, and dirflg parameters, but only the daddr parameter is required. If you don't specify a value for saddr, the starting point is "here."An address string that geolocation can understand.daddrThe destination address to be used as the destination point for directions.A complete directions request includes the saddr, daddr, and dirflg parameters, but only the daddr parameter is required.An address string that geolocation can understand.dirflgThe transport type. A complete directions request includes the saddr, daddr, and dirflg parameters, but only the daddr parameter is required. If you don't specify one of the documented transport type values, the dirflg parameter is ignored; if you don't specify any value, Maps uses the user's preferred transport type or the previous setting.sllThe search location. You can specify the sll parameter by itself or in combination with the q parameter. For example, is a valid query.A comma-separated pair of floating point values that represent latitude and longitude (in that order).sspnThe screen span. Use the sspn parameter to specify a span around the search location specified by the sll parameter. A coordinate span (see MKCoordinateSpan) denoting a latitudinal delta and a longitudinal delta.You can use the maps URL scheme to help the user:Perform a searchGet directions to a locationView a specific locationTo perform a search, supply a properly encoded URL string as the value of the q parameter. For example:To specify a location for search, you can supply a value for the near parameter, or combine the sll parameter with either the z or sspn parameters. You can also set the map type using the t parameter, as shown here:To provide navigation directions from one location to another, supply the start and destination addresses in the saddr and daddr parameters as shown below. You can also supply much more detail for the start and destination addresses than is shown here.To specify a transport type, you can use the dirflg parameter as shown here:You can omit the start address when you want to provide directions "from here." The following example shows a search from here that provides driving directions in a hybrid map.To display a specific location, use the ll parameter to center the map at the specified position as shown here:Another way to display a location is to specify an address, such as:If you use both the ll and address parameters, ll takes precedence over address. If you include a name in the value of the q parameter, Maps tries to match the name at the specified location. In today's location-aware world, users expect seamless access to place information, directions, and navigation. Whether it's through a messenger app, a local context discovery app, a logistics and transportation platform, a travel planner, or a real estate listing platform, users often need to quickly view location details or find the best route from A to B. While developers can build their own in-app experience, leveraging the comprehensive and familiar interface of Google Maps provides a superior experience. A well-structured Google Maps URL is essential for a seamless user experience. Incorrect URLs compromise this experience, directing users to the wrong location, displaying a generic map view instead of specific details, or even resulting in a broken link. This frustrates users and prevents them from achieving their goal. For example, even with a valid Maps URL, a user expecting details about a specific business might instead land on a general map view with no relevant information. See example below: This Maps URL is valid for opening Google Maps and displaying a location based on latitude and longitude. However it does not provide users with details about a specific place. Search only using latitude and longitude Seamlessly bridging your application with Google Maps through accurate URLs A Maps URL with a unique place name directs users to that place's details page on Google Maps Google Maps Platform (GMP) offers two primary methods for constructing accurate URLs: Places API(New), which requires an API key, and Maps URLs, which are free and don't require a key. The following solutions address different scenarios and use cases: The GMP Places API(New) returns a comprehensive set of information about a specific place. When requesting the googleMapsUri or googleMapsLinks field (by specifying it in field mask), the API response includes a place object. This object contains pre-formatted URLs, which can be used to directly open the corresponding view, such as place details view, in Google Maps. Example: Place Details Request curl -X GET -H 'Content-Type: application/json' \ -H "X-Goog-Api-Key: YOUR_API_KEY" \ -H "X-Goog-FieldMask: googleMapsUri,googleMapsLinks" \ Place Details Response: { "googleMapsUri": " "googleMapsLinks": { "directionsUri": " '/data=!4m7!4m6!1m1!4e2!1m2!1m1!1s0x6b12ae665e892fdd:0x3133f8d75a1ac251!3e0", "placeUri": " "writeAReviewUri": " !4m3!3m2!1s0x6b12ae665e892fdd:0x3133f8d75a1ac251!12e1", "reviewsUri": " !4m4!3m3!1s0x6b12ae665e892fdd:0x3133f8d75a1ac251!9m1!1b1", "photosUri": " !4m3!3m2!1s0x6b12ae665e892fdd:0x3133f8d75a1ac251!10e5" } } ChIJ3S-JXmauEmsRUcIaWtf4MzE in the above example is the Place ID of Sydney Opera House in the above example. A Place ID is a textual identifier that uniquely identifies a place in the Google Places database and on Google Maps. Retrieving Place IDs at zero cost To programmatically retrieve Place IDs, you can use the Places API: Text Search(ID Only) functionality. This is a zero cost method for obtaining Place IDs. Learn more about Place ID, and Place API(New) request. Places API Text Search(ID Only) Request: curl -X POST -d '{"textQuery" : "Sydney Opera House"}' \ -H 'Content-Type: application/json' -H 'X-Goog-Api-Key: YOUR_API_KEY' \ -H 'X-Goog-FieldMask: places.id' \ ' searchText' Places API Text Search(ID Only) Response: { "places": [{ "id": "ChIJ3S-JXmauEmsRUcIaWtf4MzE" }] } Place IDs can also be retrieved when a user clicks or taps a POI on maps. Find out more details about clickable POI icons(JavaScript, Android, iOS) Implementation When using the Places API, developers can simply extract the googleMapsUri or googleMapsLinks field from the response and use it to launch the corresponding view in Google Maps app or in a browser if the app is not installed. Feature Description directionsUri Link to open Google Maps to display directions from the user's current location to this place placeUri Link to open Google Maps to the place details page for this place writeAReviewUri Link to open Google Maps to the review writing page for this place reviewsUri Link to open Google Maps to the reviews page for this place photosUri Link to open Google Maps to the photos page for this place Check out the developer guidance and try the feature today. Utilizing the Maps URLs Using Maps URLs, you can build a universal, cross-platform URL to launch Google Maps and perform searches, get directions and navigation, and display map views and panoramic images. The URL syntax is the same regardless of the platform in use. You don't need a Google API key to use Maps URLs. The map actions available are: The Search function launches a Google Maps app or in a browser if the app is not installed that displays a pin for a specific place, or performs a general search and launches a map to display the results. The Directions function launches a Google Maps app or in a browser if the app is not installed that displays directions between points or enables turn-by-turn navigation on Google Maps for mobile devices. The Street View panorama function lets you launch a viewer to display Street View images as interactive panoramas. Visit GMP Maps URLs developer documentation to explore more functions and examples. Here we dive into two key functionalities of Maps URLs: Displaying place details on Google Maps: This section explains how to construct URLs that will display details for a specific location on Google Maps. It also details how to work with places that have non-unique names by using Place IDs and precise queries. Providing Directions using Maps URLs: This section explains how to create URLs that provide directions between locations, including directions with multiple waypoints and turn-by-turn navigation. Displaying place details on Google Maps The Search function takes two parameters to complete the places search, which are query(required) and query_place_id(optional). The query parameter is required for all search requests. It accepts a place name, or comma-separated latitude/longitude coordinates, or a general search term. The Search URL structure: Scenario 1: Displaying place details for a unique place name In this example, only the place name is specified. This URL opens the details page for the Sydney Opera House. Search place name and display place details Now, consider a place with a non-unique name. What happens when you search only by this non-unique name? See next scenaro. Scenario 2: Search places for a non-unique place name Since the place name is not unique, this URL opens a list of nearby 7-Eleven locations within the viewport. Users can then choose a specific store to see its details. Place list page for searching non-unique name To avoid the list of locations and directly access a specific details page, you can use a more precise method. See next example. Scenario 3: Displaying place details for a non-unique place name When dealing with common place names, a simple name search often returns a list of locations. To directly link to a specific details page, you can use one of the following methods: Method 1: Using a precise query with Place Name and Address In this URL, the query parameter is formatted as place name, address. This helps narrow down the search and directly link to the intended location. Method 2: Using the Place ID Place IDs uniquely identify a place in the Google Places database and on Google Maps. Here, ChIJGcmcg7ZC1moRAOacd3HoEwM is the unique Place ID for the specific place. The query parameter is still required but will be only used if Google Maps cannot find the place ID. Scenario 4: Displaying place details using latitude and longitude coordinates as well as the place ID Using the Place ID ensures Google Maps shows detailed place information. Place Detail page using latitude and longitude as well as Place ID Retrieving Place IDs at zero cost To programmatically retrieve Place IDs, you can use the Places API: Text Search(ID Only) functionality. This is a zero cost method for obtaining Place IDs. Learn more about Place ID, and Place API(New) request. Places API Text Search(ID Only) Request: curl -X POST -d '{"textQuery" : "Sydney Opera House"}' -H 'Content-Type: application/json' -H 'X-Goog-Api-Key: YOUR_API_KEY' -H 'X-Goog-FieldMask: places.id' ' searchText' Places API Text Search(ID Only) Response: { "places": [{ "id": "ChIJ3S-JXmauEmsRUcIaWtf4MzE" }] } Place IDs can also be retrieved when a user clicks or taps a POI on maps. Find out more details about clickable POI icons(JavaScript, Android, iOS) Conclusion Providing accurate place details is critical for a positive experience. To ensure users land on the correct place details page, construct the Search URLs using one of the following recommended formats: query=PLACE_NAME, ADDRESS query=PLACE_NAME&query_place_id=PLACE_ID Avoid using only latitude/longitude coordinates in the query parameter when your goal is to display a specific place's details. Formats like query=latitude,longitude, query=PLACE_NAME,latitude,longitude, or query=ADDRESS,latitude,longitude will not consistently lead to the desired place details page. Instead, it will show the latitude and longitude of the location. Categorical search using Maps URLs In a categorical search, you pass a general search term, and Google Maps attempts to find listings that match your criteria near the location you specify. If no location is specified, Google Maps attempts to find listings nearby your current location. Scenario 1: Nearby places search Categorical search - neaby places The Directions function displays the path between two or more specified points on the map, as well as the distance and travel time. It offers developers more control over the directions provided. GMP's Maps URLs Directions documentation provides detailed instructions on constructing URLs for customized directions. The Directions URL structure: Scenario 1: Finding the best route from the user's current location to a destination This URL opens Google Maps and displays driving directions from the user's current location. In this URL, the origin is omitted. When origin is omitted, the path defaults to the most relevant starting location, such as device location, if available. If none, the resulting map provides a form to allow a user to enter the origin. The value of origin and destination can be either a place name, address, or comma-separated latitude/longitude coordinates. The travelmode is an optional parameter. It defines the method of travel. This parameter can be set as: driving walking bicycling two-wheeler transit If no travelmode is specified, the Google Map shows one or more of the most relevant modes for the specified route and/or user preferences. Developers can also specify Place IDs by using origin_place_id parameter and destination_place_id. Using a Place ID is the best guarantee that you will link to the right place. Retrieving Place IDs at zero cost To programmatically retrieve Place IDs, you can use the Places API: Text Search(ID Only) functionality. This is a zero cost method for obtaining Place IDs. Learn more about Place ID, and Place API(New) request. Places API Text Search(ID Only) Request: curl -X POST -d '{"textQuery" : "Sydney Opera House"}' -H 'Content-Type: application/json' -H 'X-Goog-Api-Key: YOUR_API_KEY' -H 'X-Goog-FieldMask: places.id' ' searchText' Places API Text Search(ID Only) Response: { "places": [{ "id": "ChIJ3S-JXmauEmsRUcIaWtf4MzE" }] } Place IDs can also be retrieved when a user clicks or taps a POI on maps. Find out more details about clickable POI icons(JavaScript, Android, iOS) Directions from user's current location Scenario 3: Providing Turn-by-Turn Navigation Setting dir_action=navigate in the URL will launch Google Maps in turn-by-turn navigation mode if the user's current location(device location) is available and used as the origin (either explicitly provided or implicitly used when the origin parameter is omitted). Otherwise, a route preview will be displayed. Turn-by-turn navigation is launched when dir_action=navigate is set and: The origin is specified and it is close to the user's current location The origin is omitted and the user's current location is available Route preview is launched when: dir_action=navigate is not set dir_action=navigate is set and the origin is specified, and the origin is not close to the user's current location dir_action=navigate is set and the origin is omitted, and the user's current location is unavailable Note that navigation is not available on all Google Maps products such as Google Maps Web, and/or between all destinations. In those cases this parameter will be ignored. Turn-by-Turn Navigation Route Preview Conclusion By correctly constructing the Maps URLs, you can ensure users get the right information they need quickly and efficiently. Always specify the destination, and use Place ID whenever possible for guaranteed accuracy When the goal is to provide immediate navigation, include dir_action=navigate parameter to trigger turn-by-turn navigation. Navigation will start from the user's current location if the device location is available and used as the origin (either explicitly set or omitted) Choosing the right approach for your application You have two primary options: leveraging the pre-formatted URLs provided by the Places API or manually constructing Maps URLs in your application. Each approach has its own set of advantages and disadvantages. Places API: The googleMapsUri and googleMapsLinks fields in the Place Details response provide ready-to-use URLs. This reduces development time and minimizes the risk of errors in URL formatting. Provides less control over the configuration of directions. While googleMapsLinks offers basic directions, it doesn't support waypoints or advanced customization. Also, it is relatively less straightforward to trigger turn-by-turn navigation directly. Maps URLs: Offers greater flexibility and control. Developers can construct URLs to display place details and configure various aspects of directions, including adding waypoints, specifying travel modes, and initiating turn-by-turn navigation. Requires a deeper understanding of the URL parameters and structure. Manual construction increases the potential for errors if not done carefully. Improving Maps URLs with UTM parameters To help Google better understand how developers are integrating Maps URLs and to ensure optimal performance, we encourage you to include UTM tracking parameters in your URL construction. By adding utm_source and utm_campaign parameters, you provide valuable data that allows us to analyze usage patterns and improve the Maps URLs product. For the utm_source parameter, use the name of your application. The utm_campaign parameter should reflect the user's intended action, such as "location_sharing," "place_details_search," or "directions_request". For example, a URL with UTM parameters might look like this: Consistently using these parameters helps us identify areas for improvement, troubleshoot issues more effectively, and ultimately deliver a better experience for all users. Next Steps Suggested further reading: Maps URL Place IDs Places API - Link to Google Maps Routes API Contributors Principal authors: Teresa Qin | Google Maps Platform Solutions Engineer If working with Basic4Android and looking for an easy fix to the problem, try this it works both Google maps and Openstreet even though OSM creates a bit of a messy result and thanx to [yndolok] for the google marker GooglemLoc=" &[Latitude]&"+"&[Longitude]&"/@"&[Latitude]&","&[Longitude]&",15z" GooglemRute=" &[FrmLatt]&","&[FrmLong]&"/"&[ToLatt]&","&[FrmLong]&"/@"&[ScreenX]&","&[ScreenY]&",14z/data=!3m1!4b1!4m2!4m1!3e0?hl=en" 'route ?hl=en OpenStreetLoc=" 16/"&[Latitude]&"/"&[Longitude]&"&layers=N" OpenStreetRute=" Last week at Google I/O we announced Google Maps URLs, a new way for developers to link directly to Google Maps from any app. Over one billion people use the Google Maps apps and sites every month to get information about the world, and now we're making it easier to leverage the power of our maps from any app or site. Why URLs?Maps can be important to help your users get things done, but we know sometimes maps don't need to be a core part of your app or site. Sometimes you just need the ability to complete your users' journey—including pointing them to a specific location. Maybe they're ready to buy from you and need to find your nearest store, or they want to set up a meeting place with other users. All of these can be done easily in Google Maps already.What you can do is use Google Maps URLs to link into Google Maps and trigger the functionality you or your users need automatically. Google Maps URLs are not new. You've probably noticed that copying our URLs out of a browser works—on some platforms. While we have Android Intents and an iOS URL Scheme, they only work on their native platforms. Not only is that more work for developers, it means any multi-user functionality is limited to users on that same platform.Cross platformSo to start, we needed a universal URL scheme we could support cross-platform—Android, iOS, and web. A messaging app user should be able to share a location to meet up with their friend without worrying about whether the message recipient is on Android or iOS. And for something as easy as that, developers shouldn't have to reimplement the same feature with two different libraries either.So when a Google Maps URL is opened, it will be handled by the Google Maps app installed on the user's device, whatever device that is. If Google Maps for Android or iOS is available, that's where the user will be taken. Otherwise, Google Maps will open in a browser.Easy to useGetting started is simple—just replace some values in the URL based on what you're trying to accomplish. That means we made it easy to construct URLs programmatically. Here are a few examples to get you started:Say someone has finished booking a place to stay and need figure out how to get there or see what restaurants are nearby: The query parameter does what it says: plugs a query in. Here we've specified a place, but if you do the same link with no location it will search near the user clicking it. Try it out: click here for sushi near you.This is similar to our query above, but this time we got back a single result, so it gets additional details shown on the page:google.com/maps/search/?api=1&query=shoreline+amphitheatre The api parameter (mandatory) specifies the version of Maps URLs that you're using. We're launching version 1.Or if a user has set up their fitness app and want to try out a new route on their bike:www.google.com/maps/dir/?api=1&destination=stevens+creek+trail&travelmode=bicycling&dir_action=navigate We can specify the travelmode to bicycling, destination to a nearby bike trail, and we're done!And we can also open StreetView directly with a focus of our choice to give a real sense of what a place is like:www.google.com/maps/@?api=1&map_action=pano&viewpoint=36.0665,-112.0906&heading=85&pitch=10&fov=75 The viewpoint is a LatLng coordinate we want to get imagery for, and heading, pitch, and fov allows you to specify exactly where to look.Need more functionality?Google Maps URLs are great to help your users accomplish some tasks in Google Maps. However, when you need more flexibility, customization, or control, we recommend integrating Google Maps into your app or site instead. This is where our more powerful Google Maps APIs come into play. With our feature-rich range of APIs, you can access full functionality and can control your camera, draw shapes on the map, or style your maps to match your apps, brand, or just for better UI. And if you want to go beyond the map we have metadata on Places, images, and much more.Learn moreWhen you're happy to delegate the heavy lifting and make use of the Google Maps app for your needs, Maps URLs are for you. Check out our new documentation.Thank you for using Google Maps URLs and the Google Maps APIs! Be sure to share your feedback or any issues in the issue tracker. This page goes through the basic, required steps to start using any of the APIs and SDKs on Google Maps Platform. Quickstart If you have never used the Google Cloud console to create a billing account or a project, click the Get Started button that links to an interactive setup experience in the Cloud console for new users: Get Started Alternatively, you can also complete the required setup steps by clicking through the following tabs: In the Google Cloud console, on the project selector page, click Create Project to begin creating a new Cloud project. Go to the project selector page Make sure that billing is enabled for your Cloud project. Confirm that billing is enabled for your project. Note: While you must set up a billing account to set up a Cloud project, actual usage of the Embed API is available at no charge. Google Cloud offers a $0.00 charge trial. The trial expires at either end of 90 days or after the account has accrued $300 worth of charges, whichever comes first. Cancel anytime. For more information, see Billing account credits and Billing. gcloud projects create "PROJECT" Read more about the Google Cloud SDK , Cloud SDK installation , and the following commands: To use Google Maps Platform, you must enable the APIs or SDKs you plan to use with your project. Note that some integrations require you to enable multiple APIs/SDKs. If you are not sure which APIs or SDKs to enable, try using the API Picker, or consult the documentation for the API/SDK you want to use. To enable one or more APIs or SDKs: See the Google Maps Platform APIs and SDKs that you can enable by going to the Maps API Library page in the Cloud console: Go to the Maps API Library page Click the API or SDK you want to enable. If the button says ENABLE, click the button to enable the API or SDK. If the button says MANAGE, the API or SDK is already enabled and you don't need to do anything further. Clicking either button will display the dashboard for the API or SDK. (Click the DISABLE button to remove the API or SDK from this project.) The following command enables all of the Maps, Routes, and Places APIs: gcloud services enable \ --project "PROJECT" \ "addressvalidation.googleapis.com" \ "areainsights.googleapis.com" \ "tile.googleapis.com" \ "aerialview.googleapis.com" \ "elevation-backend.googleapis.com" \ "routes.googleapis.com" \ "geocoding-backend.googleapis.com" \ "geolocation.googleapis.com" \ "maps-android-backend.googleapis.com" \ "maps-backend.googleapis.com" \ "maps-embed-backend.googleapis.com" \ "maps-ios-backend.googleapis.com" \ "mapsplatformdatasets.googleapis.com" \ "places-backend.googleapis.com" \ "roads.googleapis.com" \ "routeoptimization.googleapis.com" \ "static-maps-backend.googleapis.com" \ "street-view-image-backend.googleapis.com" \ "timezone-backend.googleapis.com" Read more about the Google Cloud SDK , Cloud SDK installation , and the following commands: gcloud services enable gcloud services disable Use this command to enable the Environment APIs: gcloud services enable \ --project "PROJECT" \ "airquality.googleapis.com" \ "solar.googleapis.com" \ "pollen.googleapis.com" Note: Enabling Places API also enables the Places Library, Maps JavaScript API, Places SDK for Android and Places SDK for iOS. This step only goes through the API Key creation process. If you use your API Key in production, we strongly recommend that you restrict your API key. You can find more information in the product-specific Using API Keys page. The API key is a unique identifier that authenticates requests associated with your project for usage and billing purposes. You must have at least one API key associated with your project. To create an API key: Go to the Google Maps Platform > Credentials page. Go to the Credentials page On the Credentials page, click Create credentials > API key. The API key created dialog displays your newly created API key. Click Close. The new API key is listed on the Credentials page under API keys. (Remember to restrict the API key before using it in production.) gcloud services api-keys create \ --project "PROJECT" \ --display-name "DISPLAY_NAME" Read more about the Google Cloud SDK , Cloud SDK installation , and the following commands: gcloud services api-keys create Once you have created a billing account and project, you are eligible for the Google Cloud $300 free trial. Each Google Maps Platform SKU provides a specific amount of free monthly usage, shown in the pricing list and described in Google Maps Platform billing. Google also provides credits for public programs such as nonprofits, startups, crisis response, news media, education, and transit agencies. Next steps Once you are set up, you can begin using the Google Maps Platform APIs and SDKs. To learn more, see the Overview, Developer, and Get Started guides for the API or SDK you are interested in. The APIs and SDKs are listed at Google Maps Platform Documentation (by product) and Google Maps Platform APIs (by platform). Except as otherwise noted, the content of this page is licensed under the Creative Commons Attribution 4.0 License, and code samples are licensed under the Apache 2.0 License. For details, see the Google Developers Site Policies. Java is a registered trademark of Oracle and/or its affiliates. Last updated 2025-06-23 UTC. [[["Easy to understand","easyToUnderstand","thumb-up"],["Solved my problem","solvedMyProblem","thumb-up"],["Other","otherUp","thumb-up"]],[["Missing the information I need","missingTheInformationINeed","thumb-down"],["Too complicated / too many steps","tooComplicatedTooManySteps","thumb-down"],["Out of date","outOfDate","thumb-down"],["Samples / code issue","samplesCodeIssue","thumb-down"],["Other","otherDown","thumb-down"]],["Last updated 2025-06-23 UTC."],[[["To start using Google Maps Platform, you must create a billing-enabled Cloud project and enable the desired APIs or SDKs."],["An API key is required to authenticate requests and track usage; it's crucial to restrict your API key for production use."],["Google offers a $300 free trial for Google Cloud Platform and a $200 monthly credit for Google Maps Platform for eligible accounts."],["This guide provides the essential setup steps; for detailed API or SDK usage, refer to the respective documentation."]]],["To begin using Google Maps Platform APIs and SDKs, first, create a new Cloud project and enable billing in the Google Cloud console or via Cloud SDK. Next, enable the desired APIs or SDKs through the Maps API Library page or by using `gcloud services enable`. Finally, create an API key in the Credentials page or using `gcloud services api-keys create`; it is highly recommended to restrict this key before production use. Users are eligible for billing credits."]] If working with Basic4Android and looking for an easy fix to the problem, try this it works both Google maps and Openstreet even though OSM creates a bit of a messy result and thanx to [yndolok] for the google marker GooglemLoc=" &[Latitude]&"+"&[Longitude]&"/@"&[Latitude]&","&[Longitude]&",15z" GooglemRute=" &[FrmLatt]&","&[FrmLong]&"/"&[ToLatt]&","&[FrmLong]&"/@"&[ScreenX]&","&[ScreenY]&",14z/data=!3m1!4b1!4m2!4m1!3e0?hl=en" 'route ?hl=en OpenStreetLoc=" 16/"&[Latitude]&"/"&[Longitude]&"&layers=N" OpenStreetRute=" If working with Basic4Android and looking for an easy fix to the problem, try this it works both Google maps and Openstreet even though OSM creates a bit of a messy result and thanx to [yndolok] for the google marker GooglemLoc=" &[Latitude]&"+"&[Longitude]&"/@"&[Latitude]&","&[Longitude]&",15z" GooglemRute=" &[FrmLatt]&","&[FrmLong]&"/"&[ToLatt]&","&[FrmLong]&"/@"&[ScreenX]&","&[ScreenY]&",14z/data=!3m1!4b1!4m2!4m1!3e0?hl=en" 'route ?hl=en OpenStreetLoc=" 16/"&[Latitude]&"/"&[Longitude]&"&layers=N" OpenStreetRute=" If working with Basic4Android and looking for an easy fix to the problem, try this it works both Google maps and Openstreet even though OSM creates a bit of a messy result and thanx to [yndolok] for the google marker GooglemLoc=" &[Latitude]&"+"&[Longitude]&"/@"&[Latitude]&","&[Longitude]&",15z" GooglemRute=" &[FrmLatt]&","&[FrmLong]&"/"&[ToLatt]&","&[FrmLong]&"/@"&[ScreenX]&","&[ScreenY]&",14z/data=!3m1!4b1!4m2!4m1!3e0?hl=en" 'route ?hl=en OpenStreetLoc=" 16/"&[Latitude]&"/"&[Longitude]&"&layers=N" OpenStreetRute=" Using Maps URLs, you can build a universal, cross-platform URL to launch Google Maps and perform searches, get directions and navigation, and display map views and panoramic images. The URL syntax is the same regardless of the platform in use. You don't need a Google API key to use Maps URLs. Universal cross-platform syntax As a developer of an Android app, an iOS app, or a website, you can construct a common URL, and it will open Google Maps and perform the requested action, no matter the platform in use when the map is opened. On an Android device: If Google Maps app for Android is installed and active, the URL launches Google Maps in the Maps app and performs the requested action. If the Google Maps app is not installed or is disabled, the URL launches Google Maps in a browser and performs the requested action. On an iOS device: If Google Maps app for iOS is installed, the URL launches Google Maps in the Maps app and performs the requested action. If the Google Maps app is not installed, the URL launches Google Maps in a browser and performs the requested action. On any other device, the URL launches Google Maps in a browser and performs the requested action. It is recommended that you use a cross-platform URL to launch Google Maps from your app or website, since these universal URLs allow for broader handling of the maps requests no matter the platform in use. For features that may only be functional on a mobile platform (for example, turn-by-turn navigation), you may prefer to use a platform-specific option for Android or iOS. See the following documentation: Launching Google Maps and performing a specific action To launch Google Maps and optionally perform one of the supported functions, use a URL scheme of one of the following forms, depending on the action requested: Search — launch a Google Map that displays a pin for a specific place, or perform a general search and launch a map to display the results: Directions — request directions and launch Google Maps with the results: Display a map — launch Google Maps with no markers or directions: ?api=1&map_action=map¶meters Display a Street View panorama — launch an interactive panorama image: ?api=1&map_action=pano¶meters Important: The parameter api=1 identifies the version of Maps URLs this URL is intended for. This parameter is required in every request. The only valid value is 1. If api=1 is NOT present in the URL, all parameters are ignored and the default Google Maps app will launch, either in a browser or the Google Maps mobile app, depending on the platform in use (for example, . Constructing valid URLs You must properly encode URLs. Note: For easier reading, this document uses unencoded URLs. You should use your platform's standard URL libraries to encode them. For example, some parameters use a pipe character (|) as a separator, which you must encode as %7C in the final URL. Other parameters use comma-separated values, such as latitude/longitude coordinates or City, State. You must encode the comma as %2C. Encode spaces with %20, or replace them with a plus sign (+). Additionally, URLs are limited to 2,048 characters for each request. Be aware of this limit when constructing your URLs. Map actions The map actions available are: search, directions, display a map, and display a Street View panorama. You specify the action in the request URL, along with required and optional parameters. As is standard in URLs, you separate parameters using the ampersand (&) character. For each action, the list of parameters and their possible values are enumerated in the Search Parameters section. Search The search action displays results for a search across the visible map region. When searching for a specific place, the resulting map puts a pin in the specified location and displays available place details. Forming the Search URL Parameters query (required): Defines the place(s) to highlight on the map. The query parameter is required for all search requests. Specify locations as either a place name, address, or comma-separated latitude/longitude coordinates. Strings should be URL-encoded, so an address such as "City Hall, New York, NY" should be converted to City+Hall%2C+New+York%2C+NY. Specify general search terms as a URL-encoded string, such as grocery+stores or restaurants+in+seattle+wa. query_place_id (optional): A place ID is a textual identifier that uniquely identifies a place. For the search action, you must specify a query, but you may also specify a query_place_id. If you specify both parameters, the query is only used if Google Maps cannot find the place ID. If you are trying to definitively link to a specific establishment, the place ID is the best guarantee that you will link to the right place. It is also recommended to submit a query_place_id when you query for a specific location using latitude/longitude coordinates. Search examples Location search In a location search, you search for a specific location using a place name, address, or comma-separated latitude/longitude coordinates, and the resulting map displays a pin at that location. These three examples illustrate searches for the same location using different location values. Example 1: Searching for the place name "Lumen Field" results in the following map: Example 2: Searching for Lumen Field using only latitude/longitude coordinates results in the following map. Notice that there is a pin in the map, but no additional place information is provided on the map or in the side panel: Example 3: Searching for Lumen Field using latitude/longitude coordinates as well as the place ID results in the following map: Example 4: Searching for the place name "Starbucks" results in the following map: Example 5: Searching for a specific Starbucks using place name and place ID results in the following map: When your goal is to display a specific place's details, construct the Search URLs using one of the following recommended formats: query=PLACE_NAME,ADDRESS query=PLACE_NAME&query_place_id=PLACE_ID To programmatically retrieve Place IDs, you can use the Places API: Text Search(ID Only) features. This is a non-cost method for obtaining Place IDs. Learn more about Place ID, and Place API(New) request. Place IDs can also be retrieved when a user clicks or taps a POI on maps. Find out more details about clickable POI icons (JavaScript, Android, iOS) Categorical search In a categorical search, you pass a general search term, and Google Maps attempts to find listings that match your criteria near the location you specify. If no location is specified, Google Maps attempts to find listings nearby your current location. If you prefer to provide a location for a categorical search, include the location in the general search string (for example, pizza+seattle+wa). In this example, a search for pizza restaurants in Seattle, WA, results in the following map: Directions Directions action displays the path between two or more specified points on the map, as well as the distance and travel time. Forming the Directions URL Parameters origin: Defines the starting point from which to display directions. Defaults to most relevant starting location, such as device location, if available. If none, the resulting map may provide a blank form to allow a user to enter the origin. The value can be either a place name, address, or comma-separated latitude/longitude coordinates. A string should be URL-encoded, so an address such as "City Hall, New York, NY" should be converted to City+Hall%2C+New+York%2C+NY. Note: This parameter is optional, unless you specify an origin_place_id in your URL. If you choose to specify an origin_place_id, you must also include an origin in the URL. origin_place_id (optional): A place ID is a textual identifier that uniquely identifies a place. If you are trying to definitively specify an establishment, using a place ID is the best guarantee that you will link to the right place. URLs that use this parameter must also include an origin. destination: Defines the endpoint of the directions. If none, the resulting map may provide a blank form to allow the user to enter the destination. The value can be either a place name, address, or comma-separated latitude/longitude coordinates. A string should be URL-encoded, so an address such as "City Hall, New York, NY" should be converted to City+Hall%2C+New+York%2C+NY. Note: This parameter is optional, unless you specify a destination_place_id in your URL. If you choose to specify a destination_place_id, you must also include a destination in the URL. destination_place_id (optional): A place ID is a textual identifier that uniquely identifies a place. If you are trying to definitively specify an establishment, using a place ID is the best guarantee that you will link to the right place. URLs that use this parameter must also include a destination. travelmode (optional): Defines the method of travel. Here's the list of options: driving walking: Prefers pedestrian paths and sidewalks, where available. bicycling: Routes through bike paths, where available. "Bicycling" refers to human-powered travel mode. two-wheeler: Routes through preferred streets for two-wheelers, where available. "Two-wheeler" refers to two-wheeled motorized vehicles, such as motorcycles. For more information, see Two-wheeler coverage. transit If no travelmode is specified, the Google Map shows one or more of the most relevant modes for the specified route and/or user preferences. dir_action=navigate (optional): Launches either turn-by-turn navigation or route preview to the specified destination, based on whether the origin is available. If the user specifies an origin and it is not close to the user's current location, or the current device location is unavailable, the map launches a route preview. If the user does not specify an origin (in which case the origin defaults to the current device location), or the origin is close to the user's current location, the map launches turn-by-turn navigation. Note that navigation is not available on all Google Maps products and/or between all destinations; in those cases this parameter will be ignored. waypoints: Specifies one or more intermediary places to route directions through between the origin and destination. Multiple waypoints can be specified by using the pipe character (|) to separate places (for example, Berlin,Germany|Paris,France). The number of waypoints allowed varies by the platform where the link opens, with up to three waypoints supported on mobile browsers, and a maximum of nine waypoints supported otherwise. Waypoints are displayed on the map in the same order they are listed in the URL. Each waypoint can be either a place name, address, or comma-separated latitude/longitude coordinates. Strings should be URL-encoded, so waypoints such as "Berlin,Germany|Paris,France" should be converted to Berlin%2CGermany%7CParis%2CFrance. Notes: Waypoints are not supported on all Google Maps products; in those cases this parameter will be ignored. This parameter is optional, unless you specify waypoint_place_ids in your URL. If you choose to specify waypoint_place_ids, you must also include waypoints in the URL. waypoint_place_ids (optional): A place ID is a textual identifier that uniquely identifies a place. Waypoint_place_ids allows you to provide a list of place IDs to match the list of waypoints. Place IDs should be listed in the same order as the waypoints, and separated using the pipe character "|" (URL-encoded as %7C). If you are trying to definitively specify certain establishments, place IDs are the best guarantee that you will link to the right places. URLs that use this parameter must also include waypoints. avoid (optional): Sets features that route should try to avoid. It can be set to one or more of these options, separated by a comma character "," (URL-encoded as %2C): Directions examples The following URL launches a map in directions mode and provides a form to allow the user to enter the origin and destination. The following example launches a map with bicycling directions from the Space Needle to Pike Place Market, in Seattle, WA. The following example launches a map with two-wheeler directions from Google Mumbai to the Mumbai International Airport. Note: The following example for a two-wheeler link works only if you're physically located in the country that supports two-wheelers. This link works in countries where two-wheelers are supported, see Two-wheeler coverage.. The following example launches a map with walking directions from Google Sydney, Australia to the Queen Victoria Building. Displaying a map The map action returns a map with no markers or directions. Forming the map URL ?api=1&map_action=map¶meters Parameters map_action=map (required): Specifies the type of map view to display. Maps and Street View share the same endpoint. To ensure a map is displayed, the map_action must be specified as map. center (optional): Defines the center of the map window, and accepts latitude/longitude coordinates as comma-separated values (for example, -33.8569,151.2152). zoom (optional): Sets the initial zoom level of the map. Accepted values are whole integers ranging from 0 (the whole world) to 21 (individual buildings). The upper limit can vary depending on the map data available at the selected location. The default is 15. basemap (optional): Defines the type of map to display. The value can be either roadmap (default), satellite, or terrain. layer (optional): Defines an extra layer to display on the map, if any. The value can be one of the following: none (default), transit, traffic, or bicycling. Map examples This example URL launches a default Google Map, centered on the user's current location. ?api=1&map_action=map The following example displays a map centered on Katoomba, NSW, Australia (at -33.712206,150.311941), and sets the optional zoom and basemap parameters. ?api=1&map_action=map¢er=-33.712206%2C150.311941&zoom=12&basemap=terrain Displaying a Street View panorama The pano action lets you launch a viewer to display Street View images as interactive panoramas. Each Street View panorama provides a full 360-degree view from a single location. Images contain 360 degrees of horizontal view (a full wrap-around) and 180 degrees of vertical view (from straight up to straight down). The pano action launches a viewer that renders the resulting panorama as a sphere with a camera at its center. You can manipulate the camera to control the zoom and the orientation of the camera. Google Street View provides panoramic views from designated locations throughout its coverage area. User contributed Photospheres and Street View special collections are also available. Forming the Street View URL ?api=1&map_action=pano¶meters Parameters map_action=pano (required): Specifies the type of view to display. Maps and Street View share the same endpoint. To ensure a panorama is displayed, the action must be specified as pano. One of the following URL parameters is also required: viewpoint: The viewer displays the panorama photographed closest to the viewpoint location, specified as comma-separated latitude/longitude coordinates (for example 46.414382,10.013988). Because Street View imagery is periodically refreshed, and photographs may be taken from slightly different positions each time, it's possible that your location may snap to a different panorama when imagery is updated. pano: The specific panorama ID of the image to display. If you specify a pano you may also specify a viewpoint. The viewpoint is only used if Google Maps cannot find the panorama ID. If pano is specified but not found, and a viewpoint is NOT specified, no panorama image is displayed. Instead, Google Maps opens in default mode, displaying a map centered on the user's current location. The following URL parameters are optional: heading: Indicates the compass heading of the camera in degrees clockwise from North. Accepted values are from -180 to 360 degrees. If omitted, a default heading is chosen based on the viewpoint (if specified) of the query and the actual location of the image. pitch: Specifies the angle, up or down, of the camera. The pitch is specified in degrees from -90 to 90. Positive values will angle the camera up, while negative values will angle the camera down. The default pitch of 0 is set based on on the position of the camera when the image was captured. Because of this, a pitch of 0 is often, but not always, horizontal. For example, an image taken on a hill will likely exhibit a default pitch that is not horizontal. fov: Determines the horizontal field of view of the image. The field of view is expressed in degrees, with a range of 10 - 100. It defaults to 90. When dealing with a fixed-size viewport, the field of view is considered the zoom level, with smaller numbers indicating a higher level of zoom. Street View examples The first two examples display a panorama of the Eiffel Tower. Example one uses only a viewpoint for the location, and sets the optional heading, pitch, and fov parameters. For comparison, example two uses a pano ID as well as the same parameters set in the first example. The third example displays an indoor panorama image. Example 1: Uses only a viewpoint to specify location. ?api=1&map_action=pano&viewpoint=48.857832%2C2.295226&heading=-45&pitch=38&fov=80 Example 2: Uses a pano ID as well as a viewpoint location. The pano ID takes precedence over the viewpoint. In this example, the panorama ID is found, so the viewpoint is ignored. Notice that the panorama image that is displayed for the pano ID is slightly different, and more recent, than the image found using only the viewpoint. ?api=1&map_action=pano&pano=tu510ie_z4ptBZYo2BGEJg&viewpoint=48.857832%2C2.295226&heading=-45&pitch=38&fov=80 Example 3: Displays a panorama of the interior of the Sarastro restaurant in London, UK, specified with a panorama ID. ?api=1&map_action=pano&pano=4U-oRQCNsC6u7r8gp02sLA Finding a panorama ID To find the ID of a specific panorama image, there are multiple methods you can use. More examples Directions examples using waypoints The following directions examples launch Google Maps and display driving directions from Paris, France to Cherbourg, France, routing through the following waypoints: City, CountryPlace NamePlace ID 1. Versailles, France The Palace of Versailles ChIJdUyx15R95kcRj85ZX8H8OAU 2. Chartres, France Chartres Cathedral ChIJKzGHdEgM5EcR_OBTT3nQoEA 3. Le Mans, France Cathedral of Saint Julian of Le Mans ChIJG2LvQNCI4kcRKXNoAsPi1Mc 4. Caen, France Caen Castle ChIJ06tnGbxCCkgRsfNjEQMwUsc In the example URLs, the waypoints are defined in different ways so you can compare the differences in the display of the waypoints on the resulting maps. Example 1: Waypoints defined as City, Country: Example 2: Waypoints defined as specific place names: Example 3: Waypoints defined as City, Country, and also provides waypoint_place_ids for a specific establishment in each waypoint: Example 4: Defines waypoints as City, Country, but lists the waypoints in a different order compared to the previous examples. Demonstrates that the map displays waypoints in the order they are listed in the URL. Map examples Displays a map with a specified basemap (satellite) and layer (transit). ?api=1&map_action=map¢er=37.7992940%2C-122.3976113&zoom=15&basemap=satellite&layer=transit Street View examples Displays a Street View panorama using a FIFE image key as the pano ID. Note that the pano ID is prefaced with F:. ?api=1&map_action=pano&pano=F:-ud6AZSB-sQg/V0cfCSV3OWI/AAAAAAAAG1U/UQEoc5sXPQEIP7zzD6SWzmKb8Vbug47uQCLIB&heading=22&pitch=-5&fov=55 Improving Maps URLs with UTM parameters To help Google better understand how developers are integrating Maps URLs and to ensure optimal performance, we encourage you to include UTM tracking parameters in your URL construction. By adding utm_source and utm_campaign parameters, you provide valuable data that allows us to analyze usage patterns and improve the Maps URLs product. For the utm_source parameter, please use the name of your application. The utm_campaign parameter should reflect the user's intended action, such as "location_sharing," "place_details_search," or "directions_request." For example, a URL with UTM parameters might look like this: When you want to share a location visually, a map is an effective way to do it. But with Google Maps Platform's mapping options and even the consumer Google Maps apps, there are several approaches to consider. Choosing the right approach for you comes with different features, costs, and customization options. In this post, we'll cover three popular options and explain when each makes sense to use. Generate a custom map link to send users to the right location on maps.google.comUse the Static Maps API for a non-interactive snapshot of a locationBuild a dynamic map with the Google Maps JavaScript APIWe'll start with the quickest, simplest map tool, then add a little more complexity with each example to show you the many ways you can customize your map.**Use map links to share locations and directions**Every day, millions of people worldwide use Google Maps to find places to go and things to do while exploring the world around them. You can also use Google Maps Platform to create links to Google Maps. You can share locations or even complete directions–and it can be as easy as typing in the name of a place.This minimalist option has the added advantage of being completely free, even at high volumes. There is no need for an API Key. And to customize, you only need to edit a query URL that redirects users to Google Maps. Mobile users are sent the location in the Google Maps app (if installed). Otherwise, the link opens in the browser. To share a map link, use the search path and include the location in the query parameter. There are various styles you can use to specify the location. For example:Note that the map links are all URL encoded, which formats spaces and other characters to easily be passed through URLs. While some browsers may encode the links automatically, it's best to perform this yourself before sharing the map link.Google Maps is often used for directions, so you can build origin and destination into your map link queries. Rather than using the query parameter, the same values can be included in the origin (where your directions begin) and destination (where your directions end) parameters. For example, let's take the common tourist jaunt from the Golden Gate Bridge to Fisherman's Wharf: can combine location types, as we've done here with coordinates as the origin and a landmark as destination. Or, try leaving the origin empty to include the user's location (or allow them to enter it themselves). You can use map links in text messages, emails, or on your website within a hyperlink. Because all map links redirect to Google Maps, they aren't the right choice to visualize locations on your own site. So let's move on and cover a couple ways to create and display a map.**Mark locations with the Maps Static API**Like the map links option, the Maps Static API also involves building a custom URL. Instead of sending the user directly to the URL, it's used as the source in an HTML image tag. This allows you to mark a location on your website, without the need to re-route to Google Maps. Once you have an API key, the base URL to start with looks like this: the question mark, you need to add these parameters (at a minimum, documentation lists more options):center: the coordinates or place name of the location where you want the map to be centeredzoom: a value between 1 (world view) and 20 (building view)size: pixel width and height of the map {horizontal_value}x{vertical_value}key: your api key, which should be protected according to these best practicesFor example, here's a map centered on San Francisco: