TUTORIAL · FEBRUARY 20, 2024

Building a full-stack web app from scratch

By Dr. Holger Flick

Maps are part of many web applications. They have become a major part of the core functionality of web sites. This tutorial series will show you how easy it is to create a web application with an interactive map and provide detailed, turn-by-turn instructions as well using a backend server.

> **Source code**
> You find the source code for this example on GitHub: <https://github.com/holgerflick/hiw.directionsweb>

Thank you to the sponsor!

This video series is sponsored by TMS Software. Their enthusiasm for Delphi made it possible for me to use all their products used in this tutorial and build real-world use cases for the community. They make it possible to offer this multi-part video series for free on YouTube.

<figure markdown> ![TMS Software](https://www.tmssoftware.com) <figcaption></figcaption> </figure>

Use cases for maps on web pages

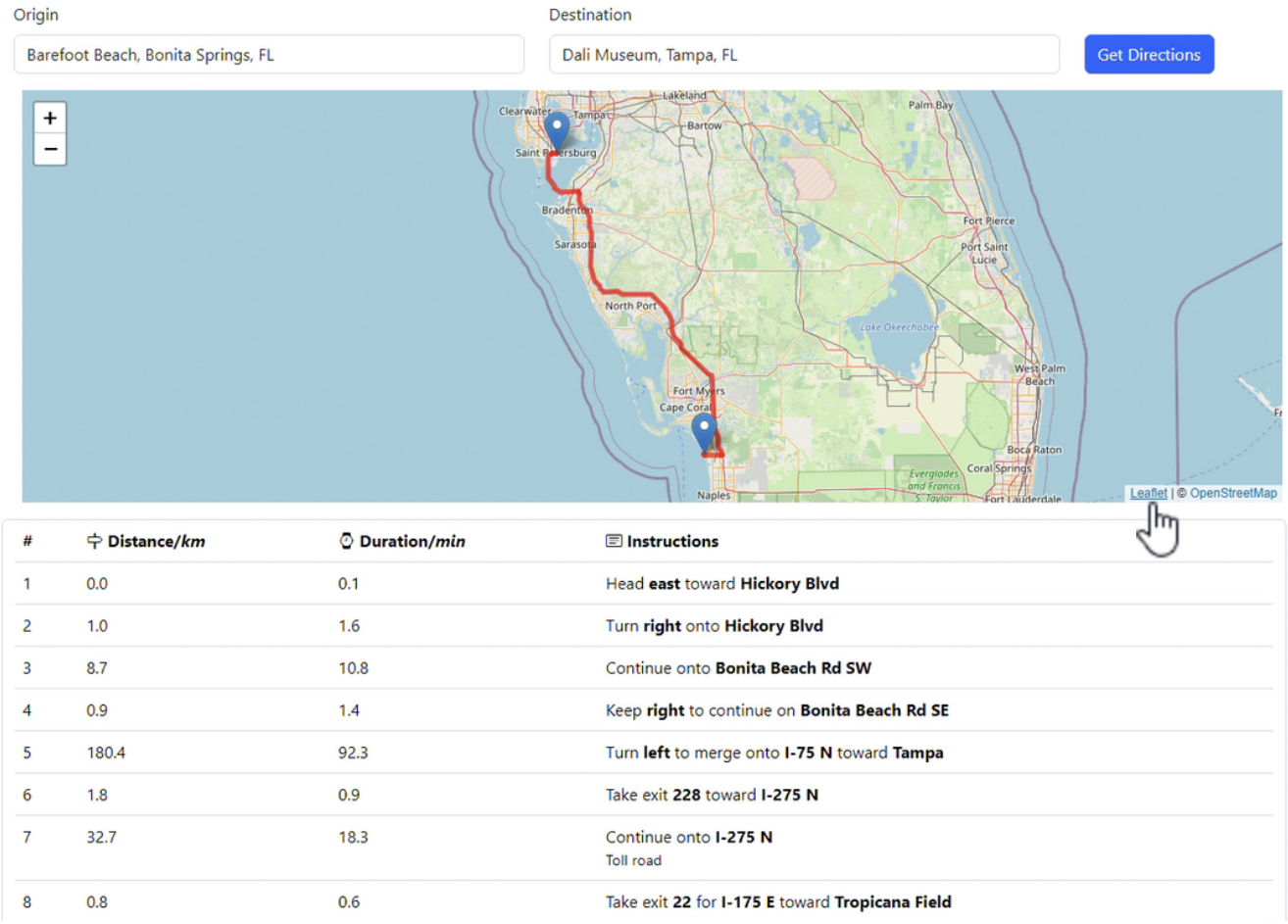
These are a few use cases:

- **Store Locators:** Retailers or businesses with multiple physical locations can use maps to help customers find nearby stores. Users can enter their location or use geolocation features to find the closest store, along with directions and additional information.
- **Travel and Tourism:** Websites related to travel and tourism frequently utilize maps to showcase destinations, points of interest, attractions, and travel routes. This can include interactive maps for planning trips, displaying tourist hotspots, or providing navigation assistance.
- **Delivery and Logistics:** Websites offering delivery services, such as food delivery or package shipping, often integrate maps to show the real-time location of deliveries, estimated arrival times, and delivery routes. This enhances transparency and allows customers to track their orders.
- **Event Planning:** Websites for events, conferences, festivals, or meetups can incorporate maps to display venue locations, parking areas, nearby accommodations, and points of interest. This helps attendees navigate to the event and explore the surrounding area.
- **Emergency Response and Planning:** Websites related to emergency services, disaster management, or public safety can use maps to display evacuation routes, emergency shelters, hazard zones, and real-time information during emergencies. This helps authorities coordinate responses and communicate critical information to the public.

In our cars, we use rather web services on our smartphones connected to Google Maps, Apple Maps, or similar.

Our use case

Looking at the possible use cases above, we create a fictional scenario that will allow your end users to pick a starting and end location for their planned travel. The web site will provide detailed driving instructions.



#	Distance/*km*	Duration/*min*	Instructions
1	0.0	0.1	Head **east** toward **Hickory Blvd**
2	1.0	1.6	Turn **right** onto **Hickory Blvd**
3	8.7	10.8	Continue onto **Bonita Beach Rd SW**
4	0.9	1.4	Keep **right** to continue on **Bonita Beach Rd SE**
5	180.4	92.3	Turn **left** to merge onto **I-75 N** toward **Tampa**
6	1.8	0.9	Take exit **228** toward **I-275 N**
7	32.7	18.3	Continue onto **I-275 N** Toll road
8	0.8	0.6	Take exit **22** for **I-175 E** toward **Tropicana Field**

Application main view of web form

The Web application is implented in *TMS WEB Core* and the map will be created **on the client** using Leaflet, a completely free mapping framework for the web. The data with the driving instructions will be provided by a **custom webservice** written in *TMS XData*. The backend will use **Google web services** to calculate the route and the detailed instructions. As we need an API key to access Google, it makes sense to store the credentials safely on the backend, completely inaccessible for the end user. Also, there are certain restrictions when it comes to retrieving driving instructions on the web client. This approach guarantees full access to all features. However, one key ingredient is that you can use **any mapping control on the client** as the backend delivers data that is not tied to a specific service. As a result, you could also implement a **Windows desktop application** that uses this data.

TMS FNC Maps is used to efficiently communicate with all Google mapping services with little code.

Content

The series is divided into nine (9) videos which are available on YouTube for free.

1. The first part introduces the use case and shows the different parts of the application. This includes both the front- and the backend. In addition, you learn how to inspect the communication of the web application with the backend.
<iframe width="560" height="315" src="https://www.youtube.com/embed/lOyR0VyFl84?si=HSUdlQscc4G71aPW" title="YouTube video player" frameborder="0" allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share" allowfullscreen></iframe>

1. Get insights into *TMS FNC Maps* and its routing features for different service providers. After learning about how to work with asynchronous web services from Google in the VCL, you will build a web service application with *TMS XData*. The focus in this part is on the types for the web service backend that will provide the data for the web application as well as the declaration of service methods.
<iframe width="560" height="315" src="https://www.youtube.com/embed/sRlxW0BMoR0?si=ql8pMHPsEhUT8M8Z" title="YouTube video player" frameborder="0" allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share" allowfullscreen></iframe>

1. Learn how to inspect and test your web services with Swagger UI. *TMS XData* generates a web interface that allows you to work with your web services without actually implementing or using a client. In addition, we will look at Fiddler (or Postman) as SwaggerUI quickl reaches its limits with the amount of data of certain requests. The web service will also be configured to work with the web application built in the later parts.
<iframe width="560" height="315" src="https://www.youtube.com/embed/GfaO_OY-bVk?si=8OCuUbRKK9HE9TwZ" title="YouTube video player" frameborder="0" allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share" allowfullscreen></iframe>

1. The server template leaves much to be desired for a bit more user comfort. This part adds a bit of eye candy as well as interaction to the main form of the XData server. Even though only server administrators in developer mode will ever encounter this, it is always good practice to learn how to start and stop an XData server and not relying on code that has been created by a wizard. VCL Action Lists will be used to simplify the interaction with the user interface.
<iframe width="560" height="315" src="https://www.youtube.com/embed/cT4i5BWY4m4?si=oICxtKPiN2NwZIwN" title="YouTube video player" frameborder="0" allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share" allowfullscreen></iframe>

1. **Please take a break before watching this part in case you binge watch this series.** Google provides asynchronous web services. In this part, we discuss options how to implement a service method in your web service that relies on another web service which will return its result "sometime in the future". One implementation is explained in detail and other options are discussed. This is the core part of the tutorial as it provides the turn-by-turn instructions to the front end.
<iframe width="560" height="315" src="https://www.youtube.com/embed/fBmf84fMke4?si=PRyFWJ3OiS8O3sjH" title="YouTube video player" frameborder="0" allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share" allowfullscreen></iframe>

1. Build the *TMS WEB Core* web application from scratch. In addition, you will learn about how to tie your source code to elements of the web design specified in HTML. Bootstrap is also used to create a responsive web application that resizes with respect to the device being used. This web application will display just fine on a mobile phone or tablet! At first, we will use *TMS FNC Maps* as well, but different controls that are part of *TMS WEB Core* are introduced and will be used in later parts.

   This part will conclude building the static parts of the user interface of the application. The data-driven parts are built next.
<iframe width="560" height="315" src="https://www.youtube.com/embed/6*Xt0ui-*

*wxSg?si=kqodx9*QLSVnZlxu" title="YouTube video player" frameborder="0" allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share" allowfullscreen></iframe>

1. Connect the user interface of the web application to the data that is provided by the backend. Learn how to build a request and process JSON information inside a TMS WEB Core web application. Display the route and markers on a map.
<iframe width="560" height="315" src="https://www.youtube.com/embed/78IYViPcFHg?si=EUh3-fhRRFuLdkaY" title="YouTube video player" frameborder="0" allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share" allowfullscreen></iframe>

1. As the last part was already pretty long, this part focuses on the last missing piece to complete the application. You will learn how to dynamically build a table with icons and formatted text based on the direction data using the web browser domain object model (DOM). At the end of this part, the web application will be fully functional.
<iframe width="560" height="315" src="https://www.youtube.com/embed/EmEAOGXCCwo?si=wpS_Dw6lHEgzSTxX" title="YouTube video player" frameborder="0" allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share" allowfullscreen></iframe>

1. In the final part, we will migrate the web application to the brand new **TMS WEB Core Leaflet control**. That way, the web client does no longer depend on TMS FNC Maps. Instead, we only use functionality provided by the Leaflet mapping framework.
<iframe width="560" height="315" src="https://www.youtube.com/embed/XEDZFQlQir0?si=dETW5y0ITxjsEibs" title="YouTube video player" frameborder="0" allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share" allowfullscreen></iframe>

INDEPENDENT · AD-FREE · READER-SUPPORTED

Free to read. Not free to make.

Every tutorial, article, and line of example code on Holger's Code is published without a paywall and without ads. This page is about what keeps it that way.

Professional work, published free

The content here is held to the same standard as professional client work: every tutorial is built as a real project, tested against current versions, documented step by step, and revisited when the frameworks move on. A single in-depth tutorial routinely takes days — and that's before the bills behind it: servers, storage, build infrastructure, licenses, and the hardware everything is verified on.

If an article or tutorial here has saved you an afternoon of debugging, taught you a technique, or helped you ship, it created real value for you. Supporting it isn't charity — it's paying for professional work you found useful, at whatever price you choose. And if you can't, keep reading anyway. That's what the free part means.

Ways to support

KO-FI

Buy me a coffee

A one-time tip — no account, no subscription, any amount.

BOOKS & COURSES

Buy the paid work

The paid tier of everything published free here — and you get something lasting in return.

SPREAD THE WORD

Share an article

Send a tutorial to a colleague, cite an article, subscribe to the RSS feed.

Nothing here goes behind a paywall, whether you chip in or not. But if something on this site earned its keep today, this is where to say so.

Support on Ko-fi —