

Developer Documentation

March 15th, 2016

Contents

[1 Introduction to Protocol Buffers 3](#_Toc445798310)

[2 Working with WorldLine Data 3](#_Toc445798311)

[2.1 Notes for Python 4](#_Toc445798312)

[3 WorldLine Static Protocol Buffer 4](#_Toc445798313)

[3.1 WorldLineStaticData Message 4](#_Toc445798314)

[3.1.1 Repeated VehicleRegistration 4](#_Toc445798315)

[3.1.2 Repeated DepartmentMotorVehicles 4](#_Toc445798316)

[3.2 VehicleRegistration Message 4](#_Toc445798317)

[3.3 DepartmentMotorVehicles Message 5](#_Toc445798318)

[4 WorldLine Sensor Protocol Buffer 5](#_Toc445798319)

[4.1 WorldLineSensorData Message 6](#_Toc445798320)

[4.1.1 Repeated TowerEvent 6](#_Toc445798321)

[4.1.2 Repeated LPREvent 6](#_Toc445798322)

[4.1.3 Repeated GPSVehicleEvent 6](#_Toc445798323)

[4.2 TowerEvent Message 6](#_Toc445798324)

[4.3 LPREvent Message 7](#_Toc445798325)

[4.4 GPSVehicleEvent Message 8](#_Toc445798326)

[5 Links and Additional Information 8](#_Toc445798327)

[Table 2 - Description of Static Data's VehicleRegistration Message 4](#_Toc444765900)

[Table 3 - Description of Static Data's DepartmentMotorVehicles Message 5](#_Toc444765901)

[Table 4 - Description of Sensor Data's TowerEvent Message 7](#_Toc444765902)

[Table 5 - Description of Sensor Data's LPREvent Message 8](#_Toc444765903)

[Table 6 - Description of Sensor Data's GPSVehicleEvent Message 8](#_Toc444765904)

# Introduction to Protocol Buffers

Developed by Google for enterprise use, Protocol Buffers are a method of serializing structured data. The serialization process involves an interface description language that describes the data structure (a .proto file), and a compiler, *protoc*, which generates source code from *proto* description files. The Protocol Buffer compiler provides source output for C#, C++, Java, Python, and more.

Protocol Buffers can be used to communicate through in-memory objects or store data on disk. WorldLine data is currently delivered as serialized data on disk. As such, this document will focus on unpacking the data from disk.

The WorldLine source code was created using Protocol Compiler Version 2.6.1, and the data written using the precompiled Java Protocol Buffer v3.0.0-beta-2.

Additional information and examples can be found in the WorldLine tools GIT repository: <https://gitlab.com/whitespace-solutions/worldline-tools>

# Working with WorldLine Data

WorldLine data is delivered on disk in the Protocol Buffer serialized data format (file extension .wl). Included with all deliveries are structure specification (.proto) files. These files are to be used with the Protocol Buffer source code compiler (protoc). The Protocol Buffer developer site contains a tutorial on running the source code compiler with a .proto file as well as examples showing the use of the resulting source code with serialized data (see Section 5 - Links and Additional Information).

There are two types of WorldLine data files. Each of these types corresponds to a specific structure description (.proto) file. The static background data is included in a single file *WorldLineStaticData.wl* (see Section 3 - WorldLine Static Protocol Buffer). Sensor data, which contains License Plate Reader data, Cell Tower data, GPS data, and Scenario Actor Ground Truth data (see Section 4 - WorldLine Sensor Protocol Buffer), is organized as multiple files per day. For each day simulated there are three sensor files with file name format Worldline\_yyyy-MM-dd\_n.wl, where yyyy-MM-dd is the simulated day and the number (n) is the nth file for that day.

## Notes for Java

Some implementations have a pre-set maximum message size that, if exceeded, cause read errors. WorldLine data can often exceed these limits. To prevent this error in Java, use the setSizeLimit function of the CodedInputStream object to increase the limit (as shown in Figure 1).

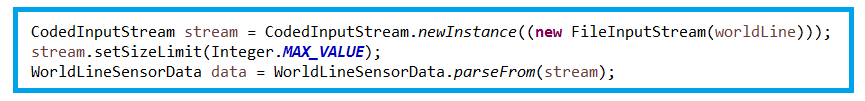


Figure 1- Increasing CodedInputStream Size Limit in Java

## Notes for C++

Similar to java, some implementations have a pre-set maximum message size that, if exceeded, can cause read errors. To prevent this error in C++, use the SetTotalBytesLimit function of the CodedInputStream class to increase the limit and the warning threshold (as shown in Figure 2).

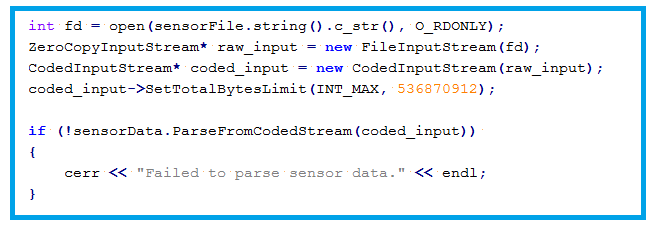


Figure 2 - Increasing CodeInputStream Size Limit in C++

## Notes for Python

The command ‘*pip install protobuff*’ currently does not work correctly. To install protobuff, use the ‘*easy\_install protobuff*’ command.

# WorldLine Static Protocol Buffer

WorldLine Static data is a single message (WorldLineStaticData) that contains “repeated messages” of two different data types – Department of Motor Vehicles and Vehicle Registration. The structure description (.proto) file for this data type is *WorldLineStatic.proto*.

## WorldLineStaticData Message

### Repeated VehicleRegistration

Each instance of VehicleRegistration corresponds to a single vehicle. If the owner is known, the vehicle maps to a person, named as the owner, from the DepartmentMotorVehicles message list.

### Repeated DepartmentMotorVehicles

Each instance of DepartmentMotorVehicles is a person and includes all known biographical information about them.

## VehicleRegistration Message

Each VehicleRegistration message contains information pertaining to a single vehicle.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Index** | **Variable Name** | **Optional/Required** | **Data Type** | **Description** |
| 1 | VehicleMake | optional | String | Make of the vehicle (eg Honda, Ford) |
| 2 | VehicleModel | optional | String | Model of the Vehicle (eg Accord, Focus) |
| 3 | VehicleYear | optional | Int32 | Year vehicle was built |
| 4 | VehicleColor | optional | String | Vehicle Color |
| 5 | VehicleIdentificationNumber | optional | String | Vehicle identification number (VIN), unique to this vehicle |
| 6 | VehicleLicensePlate | required | String | Vehicle’s License Plate |
| 7 | VehicleLicenseState | required | String | State in which the vehicle is registered |
| 8 | VehicleOwnerLastName | optional | String | Vehicle owners last name |
| 9 | VehicleOwnerFirstName | optional | String | Vehicle owners first name |
| 10 | VehicleOwnerMiddleName | optional | String | Vehicle owners middle name |
| 11 | VehicleOwnerDriversLicense | optional | String | Driver’s license number of the vehicle owner |

Table 1 - Description of Static Data's VehicleRegistration Message

## DepartmentMotorVehicles Message

Each DepartmentMotorVehicles message contains information pertaining to a single person.

DepartmentMotorVehicles contains an Enum named Sex. Sex has two options, MALE and FEMALE.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Index** | **Variable Name** | **Optional/Required** | **Data Type** | **Description** |
| 1 | LastName | required | String | Person’s last name |
| 2 | FirstName | required | String | Person’s first name |
| 3 | MiddleName | required | String | Person’s middle name |
| 4 | PersonsSex | required | Sex | Sex of the person |
| 5 | LicenseNumber | required | String | Driver’s license number. Same as VehicleOwnerDriversLicense in the VehicleRegistration message |
| 6 | BirthDate | required | Int64 | Person’s date of birth (unix / epoch time in milliseconds) |
| 7 | EyeColor | required | String | Person’s eye color (eg Brown, Blue, Green, etc) |
| 8 | HairColor | required | String | Person’s hair color (eg Blonde, Black, etc) |
| 9 | Height | required | Int32 | Height in inches |
| 10 | Weight | required | Int32 | Weight in pounds |
| 11 | Race | required | String | Person’s race (eg Hispanic, White, etc) |
| 12 | HomeAddress | required | String | Person’s Home Address |
| 13 | HomeAddressLat | required | Double | The latitude of the person’s home address in EPSG projection 4326 |
| 14 | HomeAddresLon | required | Double | The longitude of the person’s home address in EPSG projection 4326 |
| 15 | PhoneNumber | required | String | Person’s Cell Phone Number |

Table 2 - Description of Static Data's DepartmentMotorVehicles Message

# WorldLine Sensor Protocol Buffer

WorldLine sensor data is a single message (WorldLineSensorData) encompassing “repeated messages” of three different types. The distinct sensor types include Cell Tower data, License Plate Reader data, and Vehicle GPS data. The structure description (.proto) file for this data type is *WorldLineSensors.proto*.

## WorldLineSensorData Message

This message contains three repeated messages, as described above. It is the main entry point into the file format.

### Repeated TowerEvent

Each instance of TowerEvent contains one event recorded by a cell tower, including GPRS events, phone calls, and texts. For simplification purposes, each GPRS event starts when the phone connects to the tower, and ends when the phone leaves the tower (via handover or termination).

### Repeated LPREvent

Each instance of LPREvent contains one hit from a License Plate Reader, with enough information to map the vehicle back to the VehicleRegistration data included in the static data.

### Repeated GPSVehicleEvent

Each instance of GPSVehicleEvent contains a GPS record for a single vehicle, including its location and vehicle identifier to map it back to the VehicleRegistration data provided in the static data.

### Repeated PersonGroundTruthEvent

Each instance of PersonGroundTruthEvent contains the location of a given scenario actor at a given time

## TowerEvent Message

Each TowerEvent message contains one cell tower event, comprising the details of both the tower and the device.

TowerEvent contains an Enum called PhoneEventType. PhoneEventType has three options: GPRS, CALL, and SMS.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Index** | **Variable Name** | **Optional/Required** | **Data Type** | **Description** |
| 1 | TowerID | required | String | Unique identifier for the cell tower |
| 2 | TowerLat | required | Double | The latitude of the tower in EPSG projection 4326 |
| 3 | TowerLon | required | Double | The longitude of the tower in EPSG projection 4326 |
| 4 | StartTime | required | String | Date/time of the event start time (Unix / Epoch time in milliseconds) |
| 5 | EndTime | required | String | Date/time of the event end time (Unix / Epoch time in milliseconds) |
| 6 | EventType | required | PhoneEventType | Type of event, if the type is SMS or CALL, there will be a participating device. |
| 7 | IMSI | required | String | Subscriber IMSI |
| 8 | PhoneNumber | required | String | Subscriber phone number |
| 9 | CellID | required | Int32 | ID of the panel on each BTS (i.e. Tower ID). CellID is an integer in range 0-2. 0 facing north (azimuth: 0 deg), 1 facing southwest (azimuth: 225 deg), and 2 facing southeast (azimuth: 135 deg). All cell beamwidths are 120 deg. |
| 10 | CellOrientation | required | Int32 | Angle from north (counterclockwise) of the center of the cell sector. Each sector is 120 degrees wide, or 60 degrees in either direction of the CellOrientation |
| 11 | IsDeviceSender | optional | Bool | If EventType is SMS or Call, this will be set. If true, the device indicated in IMSI and PhoneNumber is the sender of the phone event if true, they are the receiver if false |
| 12 | ParticipatingDeviceNumber | optional | string | If EventType is SMS or Call, this will be set. Phone number of the participating device |

Table 3 - Description of Sensor Data's TowerEvent Message

## LPREvent Message

Each LPREvent message contains one LicensePlateReader event, including the details of the License Plate Reader and the vehicle observed.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Index** | **Variable Name** | **Optional/Required** | **Data Type** | **Description** |
| 1 | ReaderID | required | String | Unique ID of the License Plate Reader |
| 2 | ReaderLat | required | Double | The latitude of the reader in EPSG projection 4326 |
| 3 | ReaderLon | required | Double | The latitude of the reader in EPSG projection 4326 |
| 4 | ReaderOrientation | Required | Int32 | Orientation in degrees (0-360) of the license plate reader, with 0 being north and rotating counter-clockwise. |
| 5 | EventDateTime | required | Int64 | Date/time of the LPR event (Unix / Epoch time in milliseconds) |
| 6 | LicensePlate | required | String | Vehicle license plate read in the event |
| 7 | LicenseState | required | String | State the vehicle is registered in |

Table 4 - Description of Sensor Data's LPREvent Message

## GPSVehicleEvent Message

Each GPSVehicleEvent message contains a single logged GPS event for a given vehicle, including the vehicle’s information which can be mapped back to a VehicleRegistration in the static data.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Index** | **Variable Name** | **Optional/Required** | **Data Type** | **Description** |
| 1 | LicensePlate | required | String | License Plate of the vehicle containing the GPS device. |
| 2 | LicenseState | required | String | State the vehicle is registered in |
| 3 | GPSEventDateTime | required | Int64 | Date/time of the GPS event (Unix / Epoch time in milliseconds) |
| 4 | GPSLatitude | required | Double | The latitude of the GPS at the event date/time in EPSG projection 4326 |
| 5 | GPSLongitude | required | Double | The longitude of the GPS at the event date/time in EPSG projection 4326 |

Table 5 - Description of Sensor Data's GPSVehicleEvent Message

## PersonGroundTruthEvent Message

Each PersonGroundTruthEvent message contains the ground truth location at a given time for a scenario actor, and can be mapped back to a line in the DMVRecords in the static data.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Index** | **Variable Name** | **Optional/Required** | **Data Type** | **Description** |
| 1 | LastName | required | String | Last name of the scenario actor |
| 2 | FirstName | required | String | First name of the scenario actor |
| 3 | MiddleName | required | String | Middle name of the scenario actor |
| 4 | DriversLicense | required | String | Driver’s License of the scenario actor |
| 5 | PersonLatitude | required | Double | Latitude of the scenario actor at this time in EPSG projection 4326 |
| 6 | PersonLongitude | required | Double | Longitude of the scenario actor at this time in in EPSG projection 4326 |
| 7 | EventDateTime | required | Int64 | Date/Time of the given location in milliseconds since epoch time |

Table 6 - Description of Sensor Data's PersonGroundTruthEvent Message

# Links and Additional Information

WorldLine Software Tools Repository

<https://gitlab.com/whitespace-solutions/worldline-tools>

Google’s Protocol Buffers Page:

<https://developers.google.com/protocol-buffers/>

Protocol Buffer GitHub

<https://github.com/google/protobuf>

Protocol Buffer Compiler C++ pre-built Downloads

<https://repo1.maven.org/maven2/com/google/protobuf/protoc/>

Protocol Buffer Java pre-built Jar Downloads

<https://repo1.maven.org/maven2/com/google/protobuf/protobuf-java/>