Protocol Documentation

Table of Contents

Booking.proto

Booking related messages.

This file is really just an example. The data model is completely fictional.

Author: Elvis Stansvik

Booking

Represents the booking of a vehicle.

Vehicles are some cool shit. But drive carefully!

Table 1. `Booking` Fields

Field	Type	Label	Description
vehicle_id	int32	required	ID of booked vehicle.
customer_id	int32	required	Customer that booked the vehicle.
status	BookingStatus	required	Status of the booking.
confirmation_sent	bool	required	Has booking confirmation been sent?
payment_received	bool	required	Has payment been received?

BookingStatus

Represents the status of a vehicle booking.

Table 2. `BookingStatus` Fields

Field	Type	Label	Description
id	int32	required	Unique booking status ID.
description	string	required	Booking status description. E.g. "Active".

Customer.proto

This file has messages for describing a customer.

Author: Elvis Stansvik

Address

Represents a mail address.

Table 3. `Address` Fields

Field	Type	Label	Description
address_line_1	string	required	First address line.

Field	Type	Label	Description
address_line_2	string	optional	Second address line.
address_line_3	string	optional	Second address line.
town	string	required	Address town.
county	string	optional	Address county, if applicable.
country	string	required	Address country.

Customer

Represents a customer.

Table 4. `Customer` Fields

Field	Type	Label	Description
id	int32	required	Unique customer ID.
first_name	string	required	Customer first name.
last_name	string	required	Customer last name.
details	string	optional	Customer details.
email_address	string	optional	Customer e-mail address.
phone_number	string	repeated	Customer phone numbers, primary first.
mail_addresses	Address	repeated	Customer mail addresses, primary first.

Vehicle.proto

Messages describing manufacturers / vehicles.

Manufacturer

Represents a manufacturer of cars.

Table 5. `Manufacturer` Fields

Field	Type	Label	Description
id	int32	required	The unique manufacturer ID.
code	string	required	A manufacturer code, e.g. "DKL4P".
details	string	optional	Manufacturer details (minimum orders et.c.).
category	Manufacturer.Category	optional	Manufacturer category. Default: CATEGORY_EXTERNAL

Model

Represents a vehicle model.

Table 6. `Model` Fields

Field	Type	Label	Description
id	string	required	The unique model ID.
model_code	string	required	The car model code, e.g. "PZ003".
model_name	string	required	The car model name, e.g. "Z3".
daily_hire_rate_dollars	sint32	required	Dollars per day.
daily_hire_rate_cents	sint32	required	Cents per day.

Vehicle

Represents a vehicle that can be hired.

Table 7. `Vehicle` Fields

Field	Type	Label	Description
id	int32	required	Unique vehicle ID.
model	Model	required	Vehicle model.
reg_number	string	required	Vehicle registration number.
mileage	sint32	optional	Current vehicle mileage, if known.

Field	Type	Label	Description
category	Vehicle.Category	optional	Vehicle category.
daily_hire_rate_dollars	sint32	optional	Dollars per day. Default: 50
daily_hire_rate_cents	sint32	optional	Cents per day.

Table 8. `Vehicle` Nested Extensions

Extension	Type	Base	Number	Description
series	string	Model	100	Vehicle model series.

Vehicle.Category

Represents a vehicle category. E.g. "Sedan" or "Truck".

Table 9. `Vehicle.Category` Fields

Field	Type	Label	Description
code	string	required	Category code. E.g. "S".
description	string	required	Category name. E.g. "Sedan".

Manufacturer.Category

Manufacturer category. A manufacturer may be either inhouse or external.

Table 10. `Manufacturer.Category` Values

Name	Number	Description
CATEGORY_INHOUSE	0	The manufacturer is inhouse.
CATEGORY_EXTERNAL	1	The manufacturer is external.

File-level Extensions

Extension	Type	Base	Number	Description
country	string	Manufacturer	100	Manufacturer country. Default: "China"

Scalar Value Types

.proto Type	Notes	C++ Type	Java Type	Python Type
double		double	double	float
float		float	float	float
int32	Uses variable-length encoding. Inefficient for encoding negative numbers – if your field is likely to have negative values, use sint32 instead.	int32	int	int
int64	Uses variable-length encoding. Inefficient for encoding negative numbers – if your field is likely to have negative values, use sint64 instead.	int64	long	int/long
uint32	Uses variable-length encoding.	uint32	int	int/long
uint64	Uses variable-length encoding.	uint64	long	int/long
sint32	Uses variable-length encoding. Signed int value. These more efficiently encode negative numbers than regular int32s.	int32	int	int
sint64	Uses variable-length encoding. Signed int value. These more efficiently encode negative numbers than regular int64s.	int64	long	int/long
fixed32	Always four bytes. More efficient than uint32 if values are often greater than 2^28.	uint32	int	int
fixed64	Always eight bytes. More efficient than uint64 if values are often greater than 2^56.	uint64	long	int/long
sfixed32	Always four bytes.	int32	int	int
sfixed64	Always eight bytes.	int64	long	int/long
bool		bool	boolean	boolean
string	A string must always contain UTF-8 encoded or 7-bit ASCII text.	string	String	str/unicode
bytes	May contain any arbitrary sequence of bytes.	string	ByteString	str