Table of Contents

File: addressbook.proto

Message: Person

This message describes the personal contact information of a person in the database. The person contact information includes names, emails and other various contact information.

Field	Type	Rule	Description
name	string	required	This field describes the full name of the person. It should be in lastname/firstname format, and may or may not be unique.
id	int64	required	This field describes the unique ID number for this person. Seriously, this field must be globally unique, otherwise horrible things will happen.
email	string	optional	This field describes the email address of this person. If this person does not have a email addres, omit it.
phone	PhoneNumber	repeated	Nowadays, a person may have more than one phone. So this field allows for 0-n number of phones.

Enum: Person.PhoneType

This enumeration describes the different type of phone number for this person.

Element	Value	Description
MOBILE	0	Mobile cell phone type
HOME	1	Home phone type
WORK	2	Work phone type

Message: Person.PhoneNumber

This message describes the phone number of a person.

Field	Type	Rule	Description
number	string	required	The number is required, and should be in (area code)-number format.
type	PhoneType	optional	The type of this phone number. By default, pretend this is a home phone number. [default = HOME]

Message: AddressBook

Our address book file is just one of these.

Field	Type	Rule	Description
person	Person	repeated	The person list in the address book.

Scalar Value Types

A scalar message field can have one of the following types - the table shows the type specified in the .proto file, and the corresponding type in the automatically generated class:

Type	Notes	C++ Type	Java Type
double		double	double
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
int64	Uses variable-length encoding. Inefficient for encoding negative numbers - if your field is likely to have negative values, use sint64 instead.	int64	long
uint32	Uses variable-length encoding.	uint32	int
uint64	Uses variable-length encoding.	uint64	long
sint32	Uses variable-length encoding. Signed int value. These more efficiently encode negative numbers than regular int32s.	int32	int

Type	Notes	C++ Type	Java Type
sint64	Uses variable-length encoding. Signed int value. These more efficiently encode negative numbers than regular int64s.	int64	long
fixed32	Always four bytes. More efficient than uint32 if values are often greater than 2^28.	uint32	int
fixed64	Always eight bytes. More efficient than uint64 if values are often greater than 2^56.	uint64	long
sfixed32	Always four bytes..	int32	int
sfixed64	Always eight bytes.	int64	long
bool		bool	boolean
string	A string must always contain UTF-8 encoded or 7-bit ASCII text.	string	String
bytes	May contain any arbitrary sequence of bytes.	string	ByteString