RProtoBuf 0.4.9: Quick Reference Guide

Romain François Dirk Eddelbuettel Murray Stokely

March 6, 2017

example proto file `// mark as Protocol Buffers v2 format`

```
syntax = "proto2";

package tutorial;
option java_package = "com.example.tutorial";
option java_outer_classname = "AddressBookProtos";
message Person {
  required string name = 1;
  required int32 id = 2;
  optional string email = 3;
  enum PhoneType {
    MOBILE = 0;
    HOME = 1;
    WORK = 2;
  }
  message PhoneNumber {
    required string number = 1;
    optional PhoneType type = 2 [default = HOME];
  }
  repeated PhoneNumber phone = 4;
  extensions 100 to 199;
}
message AddressBook {
  repeated Person person = 1;
}
service EchoService {
  rpc Echo (Person) returns (Person);
}
```

read proto description files

```
> readProtoFiles( "somefile.proto" )
> readProtoFiles( dir = somedir )
> readProtoFiles( package = AnRPackage )
```

create a message

```
> message <- new( tutorial.Person, id = 0,
+         name = "Romain Francois",
+         email = "francoisromain@free.fr" )
```

serialize a message to a file or binary connection

```
> tf1 <- tempfile()
> message$serialize( tf1 )
> tf2 <- tempfile()
> con <- file( tf2, open = "wb" )
> message$serialize( con )
> close(con)
> message$serialize( NULL )
 [1] 0a 0f 52 6f 6d 61 69 6e 20 46 72 61 6e 63 6f
[16] 69 73 10 00 1a 16 66 72 61 6e 63 6f 69 73 72
[31] 6f 6d 61 69 6e 40 66 72 65 65 2e 66 72
```

read a message from a file or binary connection

```
> tutorial.Person$read( tf1 )
message of type 'tutorial.Person' with 3 fields set
> con <- file( tf2, open = "rb" )
> tutorial.Person$read( con )
message of type 'tutorial.Person' with 3 fields set
```

Get/Set fields

```
> email <- message$email
> message$id <- 2
> message[[ "name" ]] <- "Romain"
> id <- message[[ 2 ]]
```

Message methods

`has`	Indicates if a message has a given field.
`clone`	Creates a clone of the message
`isInitialized`	Indicates if a message has all its required fields set
`serialize`	serialize a message to a file or a binary connection or retrieve the message payload as a raw vector
`clear`	Clear one or several fields of a message, or the entire message
`size`	The number of elements in a message field
`bytesize`	The number of bytes the message would take once serialized
`swap`	swap elements of a repeated field of a message
`set`	set elements of a repeated field
`fetch`	fetch elements of a repeated field
`add`	add elements to a repeated field
`str`	the R structure of the message
`as.character`	character representation of a message
`toString`	character representation of a message (same as `as.character`)
`update`	updates several fields of a message at once
`descriptor`	get the descriptor of the message type of this message

```
> writeLines( message$toString() )
name: "Romain"
id: 2
email: "francoisromain@free.fr"
```

More info

full vignette	`vignette( "RProtoBuf")`
protobuf	http://code.google.com/p/protobuf/
RProtoBuf	http://r-forge.r-project.org/projects/rprotobuf/
mailing list	https://lists.r-forge.r-project.org/cgi-bin/mailman/listinfo/rprotobuf-yada