Style Guide

This document provides a style guide for `.proto` files. By following these conventions, you'll make your protocol buffer message definitions and their corresponding classes consistent and easy to read.

Note that protocol buffer style has evolved over time, so it is likely that you will see .proto files written in different conventions or styles. **Please respect the existing style** when you modify these files. **Consistency is key**. However, it is best to adopt the current best style when you are creating a new .proto file.

Standard file formatting

- Keep the line length to 80 characters.

- Use an indent of 2 spaces.

File structure

Files should be named `lower_snake_case.proto`

All files should be ordered in the following manner:

1. License header (if applicable)

2. File overview

3. Syntax

4. Package

5. Imports (sorted)

6. File options

7. Everything else

Packages

Package name should be in lowercase, and should correspond to the directory hierarchy. e.g., if a file is in `my/package/`, then the package name should be `my.package`.

Message and field names

Use CamelCase (with an initial capital) for message names – for example, `SongServerRequest`. Use underscore_separated_names for field names (including oneof field and extension names) – for example, `song_name`.

```
ge SongServerRequest {
uired string song_name = 1;
```

Using this naming convention for field names gives you accessors like the following:

```
st string& song_name() { ... }
d set_song_name(const string& x) { ... }
```

```
lic String getSongName() { ... }
lic Builder setSongName(String v) { ... }
```

If your field name contains a number, the number should appear after the letter instead of after the underscore. e.g., use `song_name1` instead of `song_name_1`

Repeated fields

Use pluralized names for repeated fields.

```
eated string keys = 1;

eated MyMessage accounts = 17;
```

Enums

Use CamelCase (with an initial capital) for enum type names and
CAPITALS_WITH_UNDERSCORES for value names:

```
Foo {
_UNSPECIFIED = 0;
_FIRST_VALUE = 1;
_SECOND_VALUE = 2;
```

Each enum value should end with a semicolon, not a comma. Prefer prefixing enum values
instead of surrounding them in an enclosing message. The zero value enum should have
the suffix UNSPECIFIED.

Services

If your `.proto` defines an RPC service, you should use CamelCase (with an initial capital)
for both the service name and any RPC method names:

```
ce FooService {
 GetSomething(FooRequest) returns (FooResponse);
```

Things to avoid

- Required fields (only for proto2)

- Groups (only for proto2)