syntax = "proto3";

package tensorflow;

option cc\_enable\_arenas = true;

option java\_outer\_classname = "SaverProtos";

option java\_multiple\_files = true;

option java\_package = "org.tensorflow.util";

option go\_package = "github.com/tensorflow/tensorflow/tensorflow/go/core/protobuf/for\_core\_protos\_go\_proto";

// Protocol buffer representing the configuration of a Saver.

message SaverDef {

// The name of the tensor in which to specify the filename when saving or

// restoring a model checkpoint.

string filename\_tensor\_name = 1;

// The operation to run when saving a model checkpoint.

string save\_tensor\_name = 2;

// The operation to run when restoring a model checkpoint.

string restore\_op\_name = 3;

// Maximum number of checkpoints to keep. If 0, no checkpoints are deleted.

int32 max\_to\_keep = 4;

// Shard the save files, one per device that has Variable nodes.

bool sharded = 5;

// How often to keep an additional checkpoint. If not specified, only the last

// "max\_to\_keep" checkpoints are kept; if specified, in addition to keeping

// the last "max\_to\_keep" checkpoints, an additional checkpoint will be kept

// for every n hours of training.

float keep\_checkpoint\_every\_n\_hours = 6;

// A version number that identifies a different on-disk checkpoint format.

// Usually, each subclass of BaseSaverBuilder works with a particular

// version/format. However, it is possible that the same builder may be

// upgraded to support a newer checkpoint format in the future.

enum CheckpointFormatVersion {

// Internal legacy format.

LEGACY = 0;

// Deprecated format: tf.Saver() which works with tensorflow::table::Table.

V1 = 1;

// Current format: more efficient.

V2 = 2;

}

CheckpointFormatVersion version = 7;

}