Package db

```
import "go.timothygu.me/downtomeet/server/db"
```

Overview
Index

Overview ▾

Package db implements DownToMeet server's connection to the PostgreSQL database. It also contains the database model definitions as GORM models.

Index ▾

func Get(logger FieldLogger, dsn string) (*gorm.DB, error)
func MeetupIDFromString(s string) (uint, error)
func TagIDFromString(s string) (uint, error)
func UserIDFromString(s string) (uint, error)
type Coordinates
type FieldLogger
type Logger
    func (l Logger) Error(ctx context.Context, s string, i ...interface{})
    func (l Logger) Info(ctx context.Context, s string, i ...interface{})
    func (l Logger) LogMode(_ gormlogger.LogLevel) gormlogger.Interface
    func (l Logger) Trace(ctx context.Context, begin time.Time, fc func() (string, int64), err error)
    func (l Logger) Warn(ctx context.Context, s string, i ...interface{})
type Meetup
    func (m *Meetup) IDString() string
type MeetupLocation
type Tag
    func (t *Tag) IDString() string
type User
    func (u *User) IDString() string

Package files

database.go log.go meetup.go tag.go user.go

func **Get**

```
func Get(logger FieldLogger, dsn string) (*gorm.DB, error)
```

Get returns a new gorm.DB from the given Data Source Name (DSN) connection string.

func **MeetupIDFromString**

```
func MeetupIDFromString(s string) (uint, error)
```

MeetupIDFromString reverses Meetup.IDString.

func **TagIDFromString**

```
func TagIDFromString(s string) (uint, error)
```

TagIDFromString reverses Tag.IDString.

func **UserIDFromString**

```
func UserIDFromString(s string) (uint, error)
```

UserIDFromString reverses User.IDString.

type **Coordinates**

Coordinates represent WGS 84 coordinates of the earth. It is not a database model but rather stored as a part of MeetupLocation.

```
type Coordinates struct {
    Lat, Lon *float64
}
```

type **FieldLogger**

FieldLogger is the same as log.FieldLogger, but with an additional WithContext method.

```
type FieldLogger interface {
    log.FieldLogger
    WithContext(ctx context.Context) *log.Entry
}
```

type **Logger**

Logger is an adapter from logrus' logger to GORM.

```
type Logger struct {
    Logger FieldLogger
}
```

func (Logger) Error

```
func (l Logger) Error(ctx context.Context, s string, i ...interface{})
```

Error implements GORM's logger.Interface.

func (Logger) Info

```
func (l Logger) Info(ctx context.Context, s string, i ...interface{})
```

Info implements GORM's logger.Interface.

func (Logger) LogMode

```
func (l Logger) LogMode(_ gormlogger.LogLevel) gormlogger.Interface
```

LogMode implements GORM's logger.Interface and does nothing.

func (Logger) Trace

```
func (l Logger) Trace(ctx context.Context, begin time.Time, fc func() (string,
int64), err error)
```

Trace implements GORM's logger.Interface.

func (Logger) Warn

```
func (l Logger) Warn(ctx context.Context, s string, i ...interface{})
```

Warn implements GORM's logger.Interface.

type Meetup

Meetup is the database model for a meetup.

```
type Meetup struct {
    gorm.Model
    Title             string
```

```
    Time              time.Time
    Description       string
    Tags              []*Tag `gorm:"many2many:meetup_tag;"`
    MaxCapacity       int64
    MinCapacity       int64
    Owner             uint
    Attendees         []*User        `gorm:"many2many:meetup_user_attend;"`
    Location          MeetupLocation `gorm:"embedded;embeddedPrefix:location_"`
    PendingAttendees  []*User        `gorm:"many2many:meetup_user_pending;"`
    RejectedAttendees []*User        `gorm:"many2many:meetup_user_rejected;"`
    Cancelled         bool
}
```

func (*Meetup) IDString

```
func (m *Meetup) IDString() string
```

IDString returns the meetup's ID as a string.

type MeetupLocation

MeetupLocation represents where a meetup could be held. It is not a database model but rather stored as a part of Meetup.

```
type MeetupLocation struct {
    Coordinates
    URL  string
    Name string
}
```

type Tag

Tag is the database model for tags.

```
type Tag struct {
    gorm.Model
    Name    string    `gorm:"uniqueIndex"`
    Users   []*User   `gorm:"many2many:tag_user;"`
    Meetups []*Meetup `gorm:"many2many:meetup_tag;"`
}
```

func (*Tag) IDString

```
func (t *Tag) IDString() string
```

IDString returns the tag's ID, represented as a string.

type **User**

User is the database model for users.

```
type User struct {
    gorm.Model
    Email           string `gorm:"uniqueIndex"`
    Name            string
    ContactInfo     string
    ProfilePic      *string
    FacebookID      *string     `gorm:"uniqueIndex"`
    GoogleID        *string     `gorm:"uniqueIndex"`
    Location        Coordinates `gorm:"embedded;embeddedPrefix:location_"`
    OwnedMeetups    []*Meetup   `gorm:"foreignKey:Owner"`
    Attending       []*Meetup   `gorm:"many2many:meetup_user_attend;"`
    Tags            []*Tag      `gorm:"many2many:tag_user;"`
    PendingApproval []*Meetup   `gorm:"many2many:meetup_user_pending;"`
}
```

func (*User) IDString

```
func (u *User) IDString() string
```

IDString returns the user's ID, represented as a string.