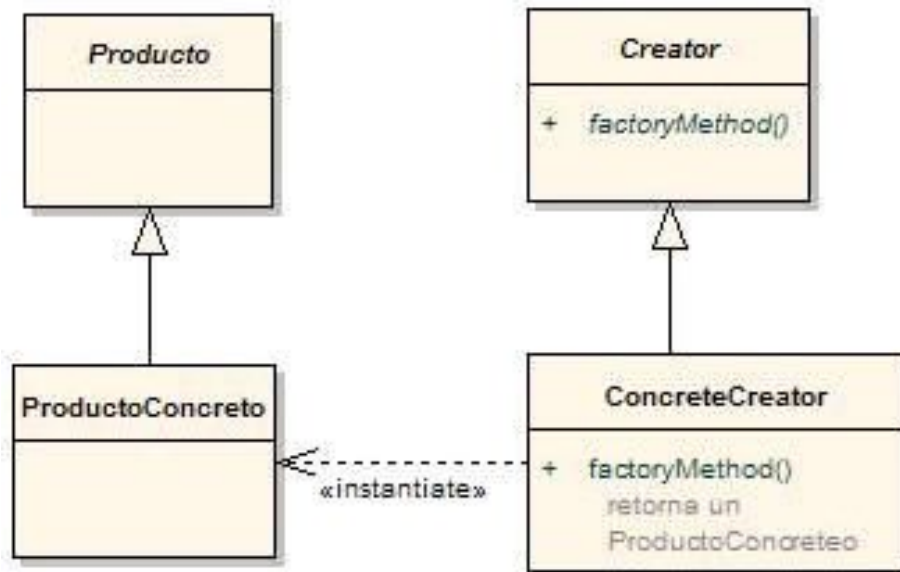


# Matron Method Factory

Angel Tomas

# Modelo Patrón Factory Method





## Cuando Usar:

- Sirven cuando no se puede anticipar el tipo de objeto que debe crear y quiere que sus subclases especifiquen dichos objetos.



## sobre el proyecto:

- Descripción del Caso: El aplicativo se trata de los tipo de seguro que maneja essalud, que maneja una carteras de seguro que podría agregarse o quitar, la intención es un manejo más flexible y ordenados de los tipos de seguro
- Está implementado en python:
- Usa base de datos sqlite3(que provee python por defecto)
- Ejecución: Para ejecutar el proyecto se requiere:
  - Ejecutar en la raíz del proyecto(donde se encuentra el archivo manage.py): `python manage.py runserver`.

# Uso del Patrón:

```
manage.py x settings.py x views.py x tests.py x models.py x
1 from django.db import models
2
3 # Modelos.
4 @| class TipoSeguro(models.Model):
5
6     descripcion = models.CharField(max_length=35)
7     estado = models.BooleanField(default=True)
8
9     def DescripcionConEstado(self):
10         cadena = "{0} - {1}"
11         strEstado = "Activo"
12
13         if self.estado == False:
14             strEstado = "Inactivo"
15
16         return cadena.format(self.descripcion, strEstado)
17
18 @| def __str__(self):
19     return self.DescripcionConEstado()
20
21
22 ##Clase Factory
23 class Seguro():
24
25     #Metodo Factory
26     def factory(tipo):
27         if tipo.pk == 1:
28             return SeguroRegular(tipo)
29         if tipo.pk == 2:
30             return SeguroIndependiente(tipo)
31         if tipo.pk == 3:
32             return SeguroRiesgoTrabajo(tipo)
33         else:
34
35 Seguro > factory() > if tipo.pk == 3
```

# Verificando los tipos de seguro:

Para agregar tipos de seguro se debe ingresar en el administrador de tablas: <http://localhost:8000/admin/> en la ppt 8,9,10 se indica las configuraciones necesarias para crear el administrador



← → ⓘ localhost:8000
<b>Lista de Tipo de Seguro de Salud - Precio</b>
<ul style="list-style-type: none"><li>• Seguro Regular - 100</li><li>• Seguro Independiente - 80,5</li><li>• Seguro de Riesgo de Trabajo - 5,0</li></ul>
Method Factory

Administración de Django

BIENVENIDO/

Sitio administrativo

AUTENTICACION Y AUTORIZACION

Grupos

+ Añadir ✎ Modificar

Usuarios

+ Añadir ✎ Modificar

GESTIONSEGUROSALUD

Tipo seguros

+ Añadir ✎ Modificar

Acciones recientes

Mis acciones

+ Seguro de Riesgo de Trabajo - Activo

Tipo seguro

+ Seguro Independiente - Activo

Tipo seguro

+ TipoSeguro object

Tipo seguro



# Creando Proyecto(python)

- Crear Proyecto:

```
django-admin startproject Patrones
```

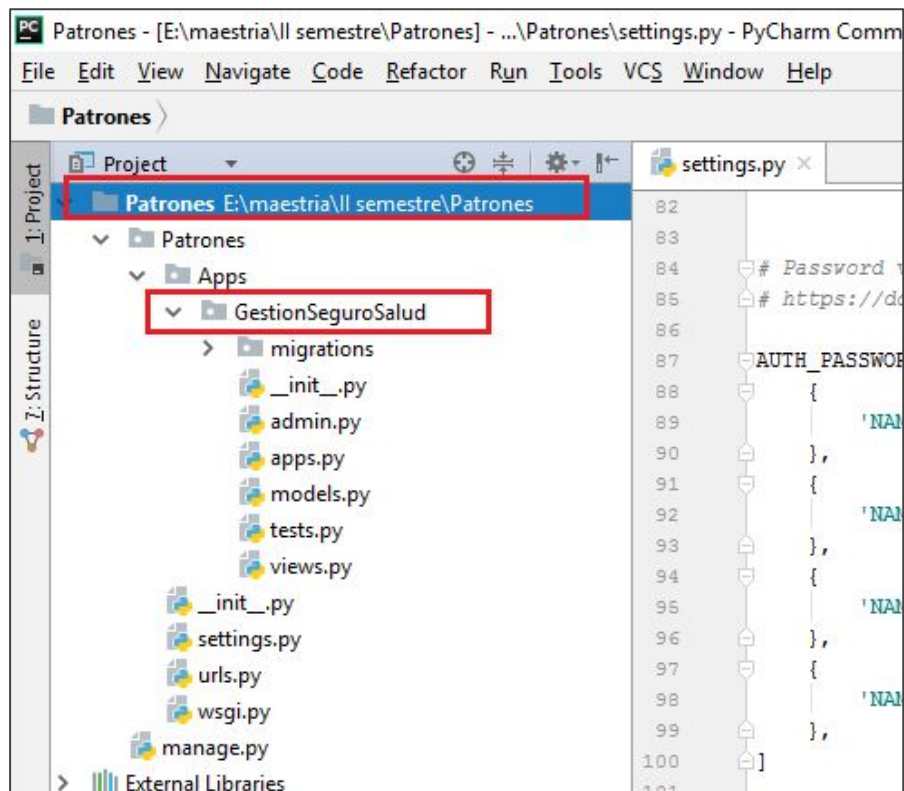
```
E:\maestria\II semestre>django-admin startproject Patrones
```

- Crear aplicativo de un proyecto:

```
django-admin startapp GestionSeguroSalud
```

```
E:\maestria\II semestre\Patrones\Patrones\Apps>django-admin startapp GestionSeguroSalud
```

# Herramienta PYCharm





# Django: Crear formularios


- Creando tablas en base de datos:

```
E:\maestria\II semestre\Patrones>python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying sessions.0001_initial... OK
```

- Mapear los modelos o tablas, se debe realizar cuando se realiza cambios.

```
E:\maestria\II semestre\Patrones>python manage.py makemigrations
Migrations for 'GestionSeguroSalud':
  Patrones\Apps\GestionSeguroSalud\migrations\0001_initial.py:
    - Create model Seguro
```

## Creando Administrador



```
E:\maestria\II semestre\Patrones>python manage.py createsuperuser

You have 1 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): GestionSeguroSalud.
Run 'python manage.py migrate' to apply them.
Username (leave blank to use 'angel'): patrones
Email address: patron@ejemplo.com
Password:
Password (again):
The password is too similar to the email address.
Password:
Password (again):
Error: Your passwords didn't match.
Password:
Password (again):
The password is too similar to the email address.
This password is too short. It must contain at least 8 characters.
Password:
Password (again):
This password is too short. It must contain at least 8 characters.
Password:
Password (again):
Superuser created successfully.
```

## Despliegue del servidor

```
E:\maestria\II semestre\Patrones>python manage.py runserver
Performing system checks...

System check identified no issues (0 silenced).

You have 1 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): GestionSeguroSalud.
Run 'python manage.py migrate' to apply them.
October 01, 2017 - 20:35:05
Django version 1.10.8, using settings 'Patrones.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

# Administrador de tablas:



# Ejecutando por Consola:

## Python Console

```
NameError: name 'mymodule' is not defined
>>> import Patrones.Apps.GestionSeguroSalud.mymodule
>>> Patrones.Apps.GestionSeguroSalud.mymodule.sum(2,1)
3
>>> import Patrones.Apps.GestionSeguroSalud.Seguro
Es Seguro Regular.
>>> Patrones.Apps.GestionSeguroSalud.Seguro.SeguroFactoryMethod("Independiente")
Traceback (most recent call last):
  File "<input>", line 1, in <module>
TypeError: object() takes no parameters
>>> Patrones.Apps.GestionSeguroSalud.Seguro.SeguroFactoryMethod.factory("Independiente")
<Patrones.Apps.GestionSeguroSalud.Seguro.Independiente object at 0x0000013816625D68>
>>> Patrones.Apps.GestionSeguroSalud.Seguro.SeguroFactoryMethod.factory("Independiente")
<Patrones.Apps.GestionSeguroSalud.Seguro.Independiente object at 0x0000013816625DD8>
>>> Patrones.Apps.GestionSeguroSalud.Seguro.SeguroFactoryMethod.factory("Independiente").drive()
Es Seguro Independiente.
>>> Patrones.Apps.GestionSeguroSalud.Seguro.SeguroFactoryMethod.factory("Independiente").drive()
Es Seguro Independiente.
>>> Patrones.Apps.GestionSeguroSalud.Seguro.SeguroFactoryMethod.factory("Regular").drive()
Es Seguro Regular.
```



# Referencias

[https://tutorial.djangogirls.org/es/django\\_start\\_project/](https://tutorial.djangogirls.org/es/django_start_project/)

<https://www.youtube.com/watch?v=9MBsaWe0SN4>

[http://apuntes-snicoper.readthedocs.io/es/latest/programacion/python/django/migraciones\\_django.html](http://apuntes-snicoper.readthedocs.io/es/latest/programacion/python/django/migraciones_django.html)

<https://www.arquitecturajava.com/usando-el-patron-factory/>