|  |
| --- |
| **MANUAL DEL PROGRAMADOR** |
| Proyecto: Sistema de votación Online |
|  |
| Repaso general de toda la estructura que compone la aplicación de forma que otros programadores se familiaricen con ella y sobre todo que sea capaz de orientarse a la hora de hacer una modificación |
|  |
|  |
| **11/10/2016** |
|  |

INDICE

TEMA PAG.

Introducción………………………………………………………………… Pág. 3

Procedimientos Almacenados ………………………………….............. Pág. 4 - 15

Estructura de Carpeta MVC………………………………………………. Pág. 16 – 18

Métodos de las Clases……………………………………………………. Pág. 19- 32

INTRODUCCION

El propósito de este manual de programador es dar a conocer al lector todos las funciones y métodos del sistema realizado. Para ello se tratará de forma amena y concisa la explicación de cada una de las Clases, Métodos y Procedimientos Almacenados..., con el fin de que otros programadores del conjunto puedan modificar a su gusto algunos de los valores y parámetros de las funciones expuestas.

También queremos proporcionar la lógica con la que se ha desarrollado la aplicación, la cual se sabe que es propia de cada programador; por lo que se considera necesaria la documentación.

**Procedimientos Almacenados**

El nombre de la base de datos es: sistemavotacion esta creada en **PostgreSQL**

**Este procedimiento almacenado es el Inicio de sesión del Administrador.**

create or replace function entrarAdministrador(

in \_dui varchar(10),

in \_contrasenia varchar(15),

out id\_usuario int,

out num\_dui varchar(10),

out contrasenia varchar(15),

out confirmacion int,

out tipo varchar(30),

out tipo\_usuario int

) returns setof record as

$body$

declare

id int;

tipo int;

texto varchar(50);

begin

return query select u.id\_usuario, ct.num\_dui, u.contrasenia, u.confirmacion, tu.tipo\_usuario, u.id\_tipo\_usuario

from usuario u inner join credencialtemporal ct

on u.id\_usuario = ct.id\_usuario

inner join tipousuario tu on tu.id\_tipo\_usuario = u.id\_tipo\_usuario

where u.id\_tipo\_usuario = 1 and ct.num\_dui = \_dui and u.contrasenia = \_contrasenia;

end;

$body$

language plpgsql;

**Inicio de sesión del Supervisor Externo**

create or replace function entrarSupervisor(

in \_identificacion varchar(10),

in \_contrasenia varchar(15),

out id\_usuario int,

out identificacion varchar(15),

out contrasenia varchar(15),

out nombre varchar(20),

out apellido varchar(20),

out sexo varchar(2),

out confirmacion int,

out tipo varchar(30),

out id\_tipo\_usuario int,

out pais varchar(15),

out organizacion varchar(15)

) returns setof record as

$body$

begin

return query select u.id\_usuario, ext.identificacion, u.contrasenia, ext.nombre, ext.apellido, ext.sexo, u.confirmacion, tu.tipo\_usuario, u.id\_tipo\_usuario, ext.pais, ext.organizacion

from usuario u inner join infosupext ext on u.id\_usuario = ext.id\_usuario

inner join tipousuario tu on tu.id\_tipo\_usuario = u.id\_tipo\_usuario

where ext.identificacion = \_identificacion and u.contrasenia = \_contrasenia;

end;

$body$

language plpgsql;

**Los usuarios principales son magistrado (2), representante del CNR (3) y director de TSE (6).**

**El administrador es quien registra a estos usuarios, siendo necesarios para el funcionamiento del sistema.**

create or replace function entrarPrincipal(

in \_dui varchar(10),

in \_contrasenia varchar(15),

out id\_usuario int,

out num\_dui varchar(10),

out contrasenia varchar(15),

out nombre varchar(20),

out apellido varchar(20),

out sexo varchar(2),

out confirmacion int,

out tipo varchar(30),

out tipo\_usuario int,

out fecha\_nac varchar(10),

out direccion varchar(60),

out id\_municipio int,

out id\_departamento int

) returns setof record as

$body$

begin

return query select u.id\_usuario, ex.num\_dui, u.contrasenia, ex.nombre, ex.apellido, ex.sexo, u.confirmacion, tu.tipo\_usuario, u.id\_tipo\_usuario, ex.fecha\_nac, ex.direccion\_especifica, mu.id\_municipio, mu.id\_departamento from usuario u

inner join credencialtemporal ct on u.id\_usuario = ct.id\_usuario

inner join excepcionusuario ex on ex.id\_usuario = u.id\_usuario

inner join municipio mu on mu.id\_municipio = ex.id\_municipio

inner join tipousuario tu on tu.id\_tipo\_usuario = u.id\_tipo\_usuario

where ex.num\_dui = \_dui and u.contrasenia = \_contrasenia and (u.id\_tipo\_usuario = 2 or u.id\_tipo\_usuario = 3 or u.id\_tipo\_usuario = 6) ;

end;

$body$

language plpgsql;

**Login para representante de partido (5), director de centro de votaciones (8),**

**Publicista (9), presidente de JRV (10) y gestor de JRV (7).**

**Son usuarios secundarios, ya que sus datos aparecen en los registros del CNR, al momento de registrar estos usuarios, solo se utiliza en número de DUI.**

create or replace function entrarSecundario(

in \_dui varchar(10),

in \_contrasenia varchar(15),

out id\_usuario int,

out num\_dui varchar(10),

out contrasenia varchar(15),

out nombre varchar(20),

out apellido varchar(20),

out sexo varchar(2),

out confirmacion int,

out tipo varchar(30),

out tipo\_usuario int,

out fecha\_nac varchar(10),

out direccion varchar(60),

out id\_municipio int,

out id\_departamento int

) returns setof record as

$body$

begin

return query select u.id\_usuario, p.num\_dui, u.contrasenia, p.nombre, p.apellido, p.sexo, u.confirmacion, tu.tipo\_usuario, u.id\_tipo\_usuario, p.fecha\_nac, p.direccion\_especifica, p.id\_municipio, m.id\_departamento from usuario u

inner join usuariopadron up on up.id\_usuario = u.id\_usuario

inner join padronelectoral p on p.num\_dui = up.num\_dui

inner join municipio m on m.id\_municipio = p.id\_municipio

inner join tipousuario tu on tu.id\_tipo\_usuario = u.id\_tipo\_usuario

where (u.id\_tipo\_usuario = 5 or (u.id\_tipo\_usuario >= 7 and u.id\_tipo\_usuario <= 10 )) and (p.num\_dui = \_dui and u.contrasenia = \_contrasenia);

end;

$body$

language plpgsql;

**Usuario que aparezca en la tabla padronelectoral, puede acceder a la papeleta de votación sin importar el privilegio que posea en el sistema con este procedimiento almacenado.**

create or replace function entrarVotante(

in \_dui varchar(10),

in \_contrasenia varchar(15),

out id\_usuario int,

out num\_dui varchar(10),

out contrasenia varchar(15),

out nombre varchar(20),

out apellido varchar(20),

out sexo varchar(2),

out confirmacion int,

out tipo varchar(30),

out tipo\_usuario int,

out fecha\_nac varchar(10),

out direccion varchar(60),

out id\_municipio int,

out id\_departamento int

) returns setof record as

$body$

begin

return query select u.id\_usuario, p.num\_dui, u.contrasenia, p.nombre, p.apellido, p.sexo, u.confirmacion, tu.tipo\_usuario, u.id\_tipo\_usuario, p.fecha\_nac, p.direccion\_especifica, p.id\_municipio, m.id\_departamento from usuario u

inner join usuariopadron up on up.id\_usuario = u.id\_usuario

inner join padronelectoral p on p.num\_dui = up.num\_dui

inner join municipio m on m.id\_municipio = p.id\_municipio

inner join tipousuario tu on tu.id\_tipo\_usuario = u.id\_tipo\_usuario

where p.num\_dui = \_dui and u.contrasenia = \_contrasenia;

end;

$body$

language plpgsql;

**Agregar Supervisores**

create or replace function agregarSupervisor(

in identificacion varchar(15),

in contrasenia varchar(15),

in nombre varchar(20),

in apellido varchar(20),

in sexo varchar(2),

in pais varchar(15),

in organizacion varchar(15)

) returns void as

$body$

begin

insert into usuario (id\_tipo\_usuario,contrasenia,confirmacion) values (4,contrasenia,0);

insert into infosupext (id\_usuario,identificacion,nombre,apellido,sexo,pais,organizacion)

values (lastval(),identificacion,nombre,apellido,sexo,pais,organizacion);

end;

$body$

language plpgsql;

Modificar Supervisor

create or replace function modificarSupervisor(

in \_id\_usuario int,

in \_identificacion varchar(15),

in \_contrasenia varchar(15),

in \_nombre varchar(20),

in \_apellido varchar(20),

in \_sexo varchar(2),

in \_pais varchar(15),

in \_organizacion varchar(15)

) returns boolean as

$body$

begin

if ((select id\_tipo\_usuario from usuario where id\_usuario = \_id\_usuario) = 4) then

update usuario set contrasenia = \_contrasenia where id\_usuario = \_id\_usuario;

update infosupext set identificacion = \_identificacion, nombre = \_nombre, apellido = \_apellido, sexo = \_sexo, pais = \_pais, organizacion = \_organizacion where id\_usuario = \_id\_usuario;

return true;

else

return false;

end if;

end;

$body$

language plpgsql;

**Magistrados, Representante CNR y director de TSE no es necesario que sus datos aparezcan en los registros del CNR para que puedan tener una cuenta de usuario, el parámetro tipo solo acepta el 2, 3 o 6 que representas a los 3 usuarios mencionados al principio**

create or replace function agregarPrincipal(

in \_num\_dui varchar(10),

in \_contrasenia varchar(15),

in \_nombre varchar(20),

in \_apellido varchar(20),

in \_fecha\_nac varchar(10),

in \_sexo varchar(2),

in \_direccion varchar(60),

in \_municipio int,

in tipo int

) returns boolean as

$body$

begin

if(tipo = 2 or tipo = 3 or tipo = 6) then

insert into usuario (id\_tipo\_usuario, contrasenia, confirmacion) values (tipo, \_contrasenia, 0);

insert into credencialtemporal (id\_usuario, num\_dui) values (lastval(), \_num\_dui);

insert into excepcionUsuario (id\_usuario,num\_dui, nombre, apellido, fecha\_nac, sexo, direccion\_especifica, id\_municipio)

values (lastval(),\_num\_dui,\_nombre,\_apellido,\_fecha\_nac,\_sexo,\_direccion,\_municipio);

return true;

else

return false;

end if;

end;

$body$

language plpgsql;

**Se agregan los ciudadanos, desde los registros del CNR a nuestra tabla.**

create or replace function agregarVotante(

in \_num\_dui varchar(10),

in \_contrasenia varchar(15),

in \_nombre varchar(20),

in \_apellido varchar(20),

in \_fecha varchar(10),

in \_sexo varchar(2),

in \_direccion varchar(60),

in \_municipio int

) returns void as

$body$

begin

insert into padronelectoral(num\_dui,nombre,apellido,fecha\_nac,sexo,direccion\_especifica,id\_municipio)

values (\_num\_dui,\_nombre,\_apellido,\_fecha,\_sexo,\_direccion,\_municipio);

insert into usuario (id\_tipo\_usuario, contrasenia, confirmacion) values (11,\_contrasenia,0);

insert into usuariopadron (id\_usuario,num\_dui) values (lastval(),\_num\_dui);

end;

$body$

language plpgsql;

Procedimiento para agregar Representante de partido (5),Gestor de jrv (7), director de centro de votaciones (8), publicista (9) y presidente de jrv (10).

Estos usuarios necesitar estar entre los registros del CNR para poder tener

una cuenta de este tipo.

create or replace function agregarSecundario(

in \_num\_dui varchar(10),

in \_contrasenia varchar(15),

in tipo int

) returns boolean as

$body$

declare

id int;

begin

if(tipo = 5 or (tipo >= 7 and tipo <= 11 )) then

/\*se comprueba que el dui exista en los registros del cnr\*/

if exists(select \* from padronelectoral where num\_dui = \_num\_dui) then

select u.id\_usuario into id from usuariopadron u where u.num\_dui = \_num\_dui;

update usuario set id\_tipo\_usuario = tipo, contrasenia = \_contrasenia where id\_usuario = id;

return true;

else

return false;

end if;

else

return false;

end if;

end;

$body$

language plpgsql;

**Modifica usuarios principales.**

create or replace function modificarPrincipal(

in \_id int,

in \_num\_dui varchar(10),

in \_contrasenia varchar(15),

in \_nombre varchar(20),

in \_apellido varchar(20),

in \_fecha\_nac varchar(10),

in \_sexo varchar(2),

in \_direccion varchar(60),

in \_municipio int,

in tipo int

) returns boolean as

$body$

begin

if(tipo = 2 or tipo = 3 or tipo = 6) then

update usuario set id\_tipo\_usuario = tipo, contrasenia = \_contrasenia where id\_usuario = \_id;

update credencialtemporal set num\_dui = \_num\_dui where id\_usuario = \_id;

update excepcionUsuario set num\_dui = \_num\_dui, nombre = \_nombre, apellido = \_apellido, fecha\_nac = \_fecha\_nac, sexo = \_sexo, direccion\_especifica = \_direccion, id\_municipio = \_municipio where id\_usuario = \_id;

return true;

else

return false;

end if;

end;

$body$

language plpgsql;

**Solo se pueden eliminar los usuarios que aun no han activado sus cuentas**

**y no hayan ingresado registros al sistema, este procedimiento se usa solo**

**en caso de que se haya cometido un error al momento de ingresar usuarios, y se quiere corregir eliminándolo.**

create or replace function eliminarSupervisor(

in \_id\_usuario int

) returns void as

$body$

begin

delete from credencialtemporal where id\_usuario = \_id\_usuario;

delete from infosupext where id\_usuario = \_id\_usuario;

delete from usuario where id\_usuario = \_id\_usuario;

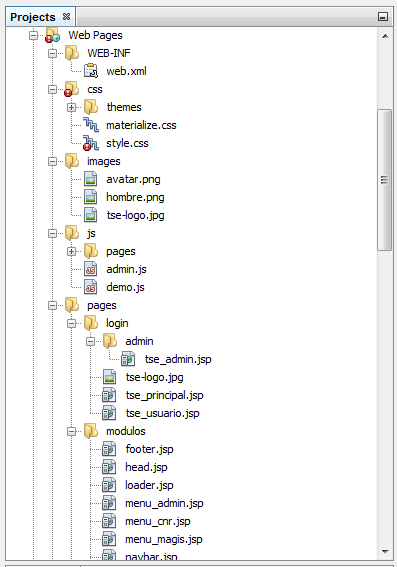
end;

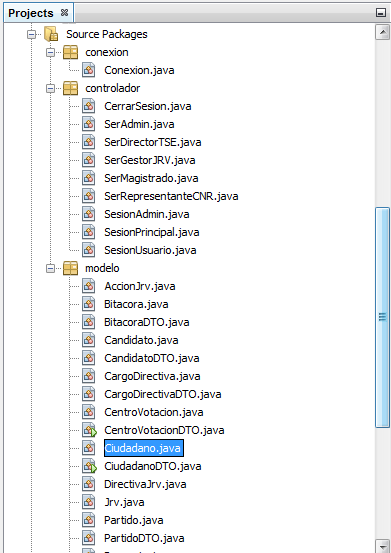
$body$

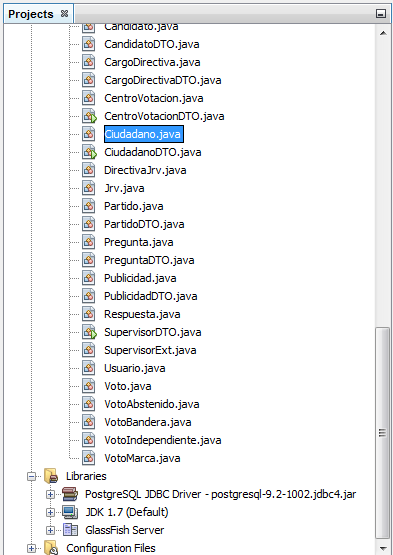
language plpgsql;

**ESTRUCTURA DE CARPETAS**

**MVC**

****

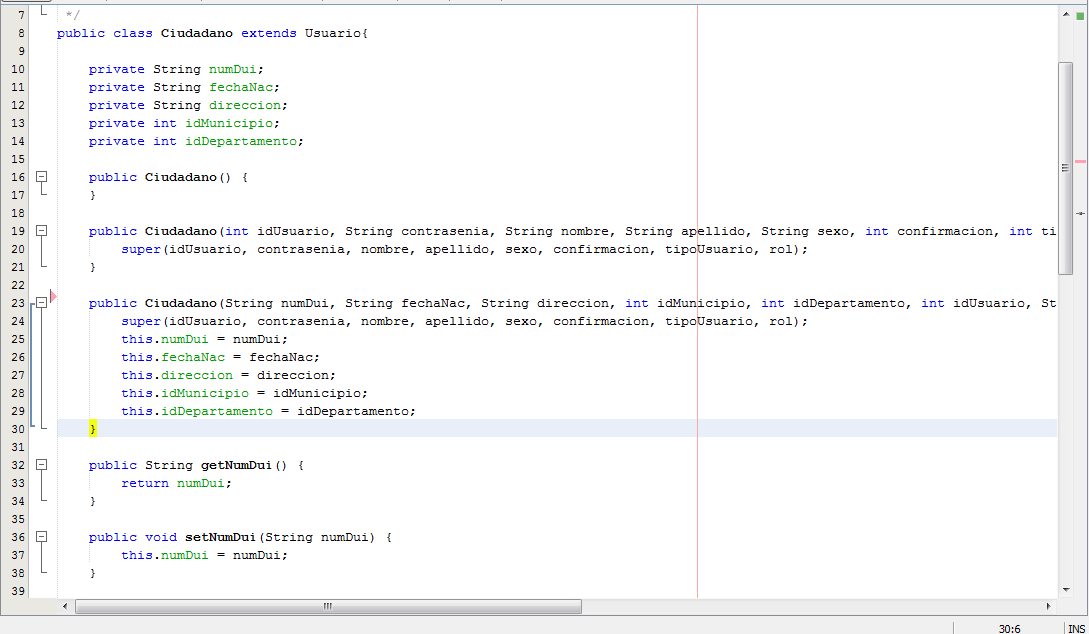




CLASES

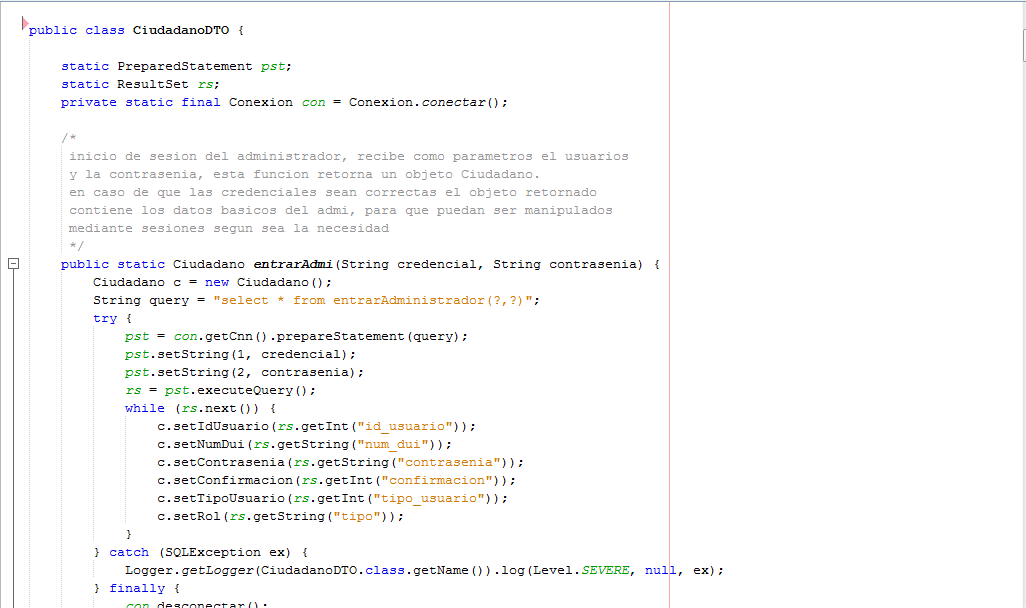
Las Clases están divididas en Clases y ClasesDTO.

Clase Ciudadano contiene los métodos getter and setter.



CuidadanoDTO

Las ClasesDTO contienen los métodos de mostrar, insertar, modificar y eliminar

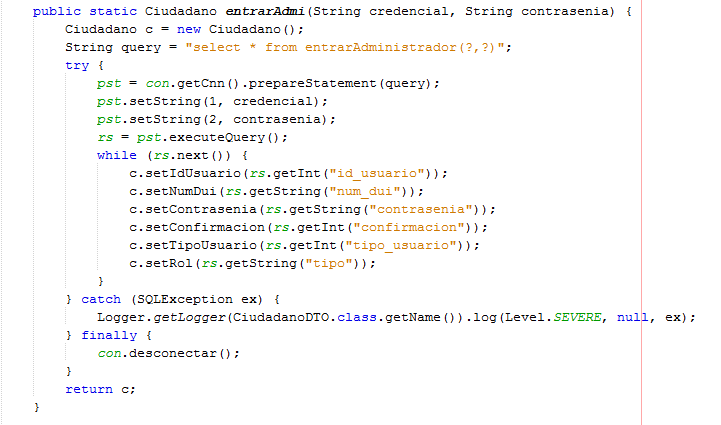


Inicio de sesión del administrador, recibe como parámetros el usuarios y la contraseña, esta función retorna un objeto Ciudadano.

En caso de que las credenciales sean correctas el objeto retornado

Contiene los datos básicos del admin, para que puedan ser manipulados

Mediante sesiones según sea la necesidad.



Esta función es utilizada por los usuarios principales del sistema que son

los magistrados, usuario del CNR y director del TSE.

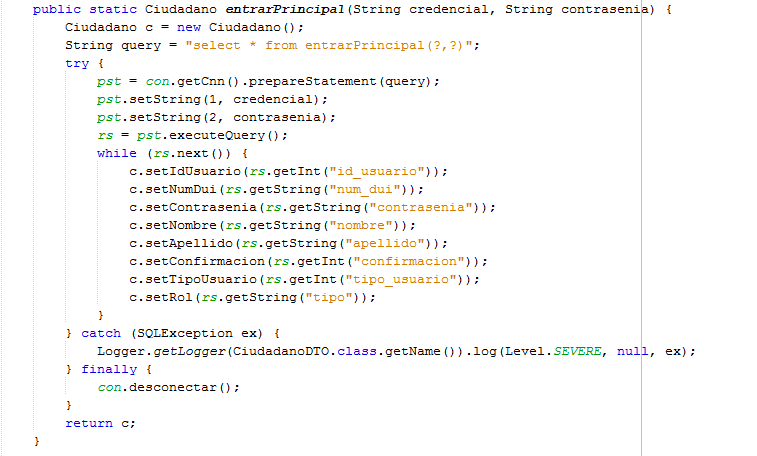
recibe como parámetros las credenciales de usuario y retorna un objeto tipo

Ciudadano con la información del usuario en caso de que las credenciales sean

correctas, en caso contrario retorna un objeto vacio.

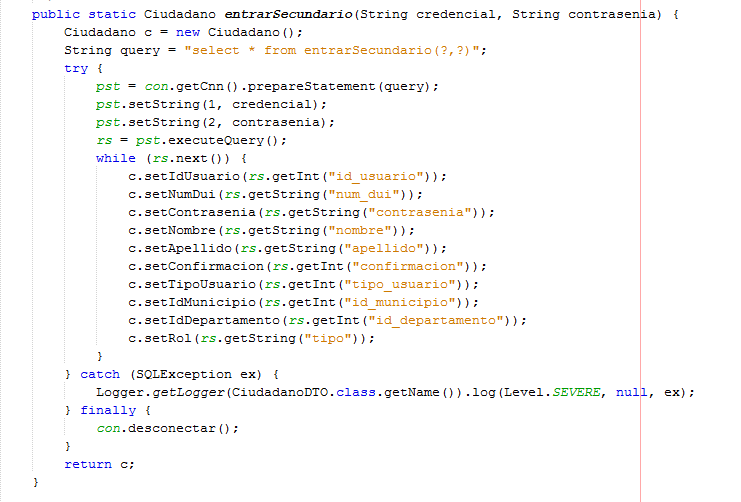
son usuarios principales porque no necesitan aparecer en la tabla con los

datos proporcionados del CNR para poder tener una cuenta en el sistema



Esta función es usada por los usuarios secundarios del sistema, que son el representante de partido, director de centro de votaciones, publicista,

Presidente de JRV y gestor de JRV. Recibe como parámetro las credenciales de usuario y retorna un objeto Ciudadano con los datos del usuario en caso de que las credenciales sean correctas, en caso contrario retorna un objeto vacio. Son usuarios secundarios ya que para que posean una cuenta es necesario que aparezcan en los registros que el CNR nos proporciona



Función en la que inician sesión todos los usuarios que aparezcan en los registros

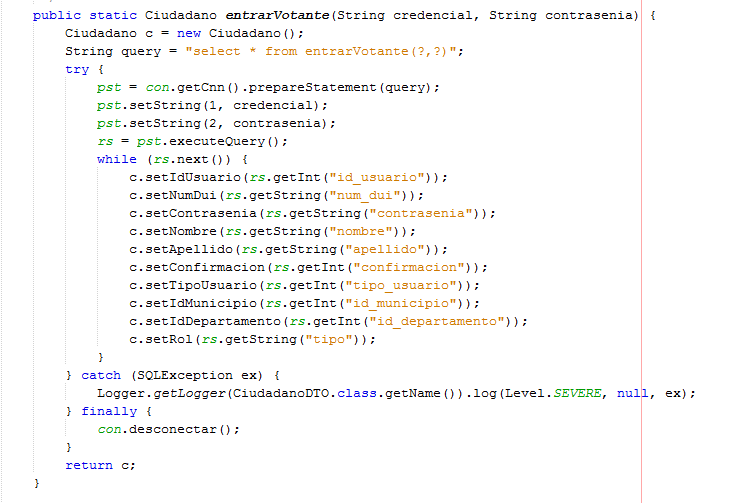
del CNR, no importa si tienen una cuenta de usuario secundario, por esta función

entrar como votantes.

se reciben como parámetros las credenciales de usuario, y se retorna un objeto

con el tipo Ciudadano que contiene la información del votante, en caso contrario

retorna el objeto vacio.



En esta función se registran todos los usuarios del sistema a excepción

del supervisor externo.

se recibe como parámetro un objeto del tipo ciudadano y según el valor

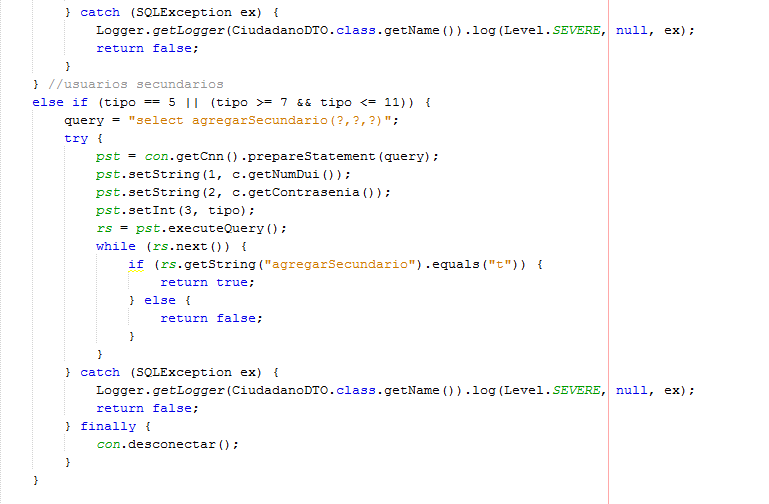
que tenga en su atributo tipoUsuario se registra como usuarioPrincipal

o usuarioSecundario.

se retorna true cuando el registro es un éxito, en caso de fallas se

retorna un false





Solo se pueden modificar usuarios principales, obteniendo los datos

del objeto que se para por parámetro y retornando true si se actualizo

Correctamente o false si hubo un error.

no se pueden modificar usuarios secundarios, ya que el procedimiento

agregarUsuario que está arriba, cuando se trata de un usuario secundario

no hace registros, solo actualiza el tipo de usuario.





Se recibe como parámetro el id de un usuario, por medio del cual se hace

una consulta retornando un objeto Ciudadano con la información de este usuario,

en caso de que no encuentre ese registro, retorna un objeto vacio

el parametro hace una diferencia cuando el usuario que se busca es administrador

retornando solo id\_usuario, dui, contraseña, confirmación y el tipo de usuario

cuando se encuentra un usuario principal, además de estos datos, retorna los datos correspondientes de la tabla excepcionusuario, con su información

cuando es un usuario secundario o un votante, retorna la misma información que un usuario principal, pero se toman de la tabla padronelectoral





El método busca a los ciudadanos que están en los registros de padronelecoral

Mediante el DUI, que es el parámetro de entrada

cuando se encuentra el registra, se retorna un obj Ciudadano con la información

del votante, en caso contrario se retorna un objeto vacio.



La función recibe como parámetro el tipo de usuario que quiere mostrar, y el departamento al que debe pertenecer, retorna un arraylist con los usuarios encontrados en la consulta cuando se ingresa 2, 3 o 6 en el primer parámetro se hace una búsqueda de los usuarios principales, en la tabla excepcionusuario cuando se ingresa 5, 7, 8, 9 o 10 se hace una búsqueda de los usuarios secundarios del sistema, sus datos toman de la tabla padronelectoral

Cuando se ingresa cualquier numero distinto a los anteriores, se hace una búsqueda de los votantes que hay registrados en el sistema, teniendo la posibilidad de filtrarlos por departamento ingresando el id\_departamento en el segundo parámetro cuando se ingresa 0 en el segundo parámetro, se omite el filtre de búsqueda de departamento





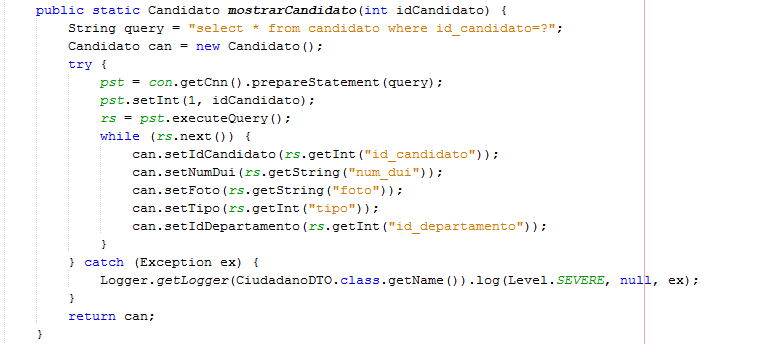
METODOS MÁS USADOS.

Mostar

se recibe como parámetro el id de un candidato, se realiza una búsqueda

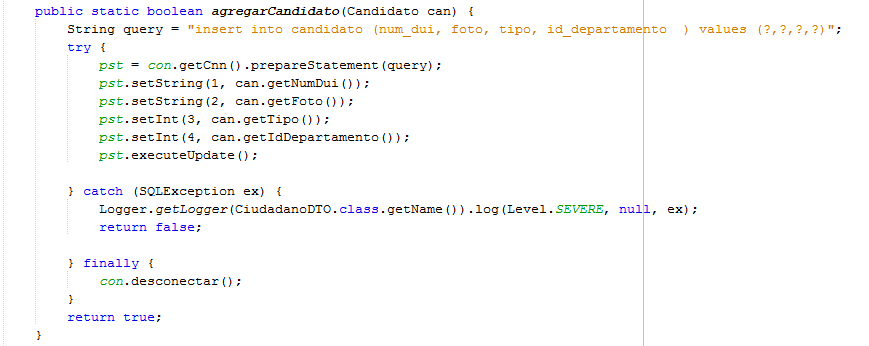
y retorna la información de este candidato en un objeto tipo Candidato

cuando no se encuentran resultados, se retorna un objeto vacio.

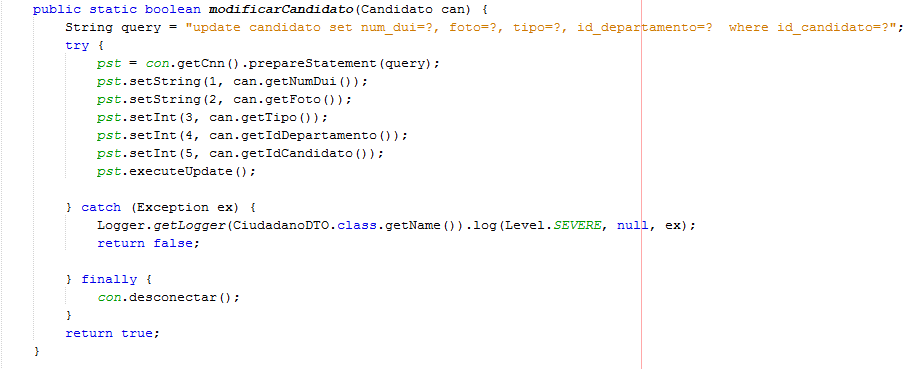


Insertar

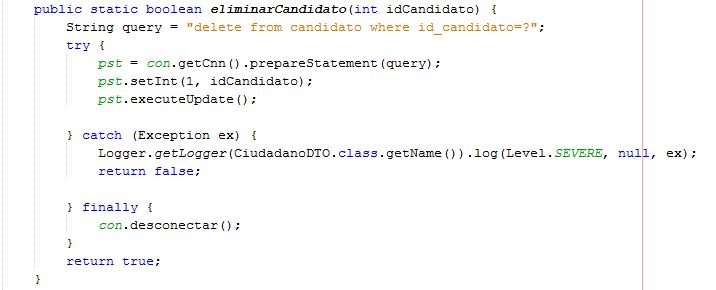
Se recibe como parámetro un objeto Candidato, en cual extrae la información de sus atributos y las inserta en un registro en la base de datos



Modificar



Eliminar



Mostrar ArrayList<>

Esta función retorna un arraylist con todos los candidatos que están registrados en el sistema.

