

15-09-2020



**Autores:**

Cristina Rodríguez Chamorro

Daniel Lanzas Pellico

Helena García Fernández

José Bennani Pareja

Juan José Lucas de la Fuente

Unai Ares Icaran

**Tutor:**

Sergio Torres Palomino

**Blockchain & BigData Canino**

ANEXO III: Chaincodes y funcionalidades del sistema

**VERSION**

**FINAL**

**Documentación del TFM**

***Máster Blockchain y Big Data.*** *Curso 2019-2020*

ÍNDICE

[**Contratos divididos por Organizaciones** 4](#_Toc50969995)

[**Federaciones Caninas** 4](#_Toc50969996)

[**Colegios de Veterinarios** 4](#_Toc50969997)

[**Contratos inteligentes** 5](#_Toc50969998)

[**Parámetros de entrada/salida de las funciones** 5](#_Toc50969999)

[**Funciones comunes de los contratos** 6](#_Toc50970000)

[**ejecutarConsulta** 6](#_Toc50970001)

[**asignarEstado** 6](#_Toc50970002)

[**borrarEstado** 6](#_Toc50970003)

[**consultarEstado** 6](#_Toc50970004)

[**consultarRangoEstados** 6](#_Toc50970005)

[**getQueryResultForQueryString** 6](#_Toc50970006)

[**PERSONAS** 7](#_Toc50970007)

[**registrarPersona** 8](#_Toc50970008)

[**registrarDocumentoIdentidad** 9](#_Toc50970009)

[**modificarNombreApellidos** 9](#_Toc50970010)

[**Funciones comunes** 10](#_Toc50970011)

[**AFIJOS** 13](#_Toc50970012)

[**registrarAfijo** 13](#_Toc50970013)

[**registrarCambioPropietario** 15](#_Toc50970014)

[**registrarCancelacionAfijo** 16](#_Toc50970015)

[**cargarDatosIniciales** 18](#_Toc50970016)

[**cargarDatosIniciales\_Propietarios** 18](#_Toc50970017)

[**consultarDatosAfijo** 19](#_Toc50970018)

[**obtenerCertificadoAfijo** 19](#_Toc50970019)

[**Funciones comunes** 19](#_Toc50970020)

[**PERROS** 22](#_Toc50970021)

[**registrarPerro** 22](#_Toc50970022)

[**registrarCambioPropietario** 25](#_Toc50970023)

[**registrarDefuncionPerro** 27](#_Toc50970024)

[**registrarReconocimientoRaza** 29](#_Toc50970025)

[**cargarDatosIniciales** 30](#_Toc50970026)

[**cargarDatosIniciales\_Propietarios** 31](#_Toc50970027)

[**consultarDatosEjemplar** 32](#_Toc50970028)

[**obtenerCertificadoRegistro** 32](#_Toc50970029)

[**obtenerPedigri** 32](#_Toc50970030)

[**registrarCesionTemporal** 32](#_Toc50970031)

[**Funciones comunes** 32](#_Toc50970032)

[**SOLICITUDES** 37](#_Toc50970033)

[**solicitarRegistrarCamada** 40](#_Toc50970034)

[**solicitarRegistrarPerro** 40](#_Toc50970035)

[**solicitarRegistrarCambioPropietarioPerro** 40](#_Toc50970036)

[**solicitarRegistrarCambioPropietarioAfijo** 40](#_Toc50970037)

[**solicitarRegistrarCancelacionAfijo** 40](#_Toc50970038)

[**validarSolicitud** 41](#_Toc50970039)

[**cargarDatosIniciales** 42](#_Toc50970040)

[**cargarDatosIniciales\_Autorizaciones** 43](#_Toc50970041)

[**Funciones complementarias** 44](#_Toc50970042)

[**Funciones comunes** 44](#_Toc50970043)

[**MICROCHIP** 48](#_Toc50970044)

[**registrarMicrochips** 48](#_Toc50970045)

[**consultarMicrochip** 49](#_Toc50970046)

[**cargarDatosIniciales** 49](#_Toc50970047)

[**Funciones comunes** 50](#_Toc50970048)

[**VACUNAS** 52](#_Toc50970049)

[**registrarVacuna** 53](#_Toc50970050)

[**obtenerCertificadoVacunaciones (pasaporte)** 54](#_Toc50970051)

[**consultarVacunaciones** 54](#_Toc50970052)

[**cargarDatosIniciales** 55](#_Toc50970053)

[**cargarDatosIniciales\_VacunasPerrosProteccion** 55](#_Toc50970054)

[**cargarDatosIniciales\_VacunasProteccion** 56](#_Toc50970055)

[**Funciones comunes** 57](#_Toc50970056)

[**PERFILES** 59](#_Toc50970057)

[**registrarPerfilPersona** 59](#_Toc50970058)

[**cancelarPerfilPersona** 60](#_Toc50970059)

[**cargarDatosIniciales** 62](#_Toc50970060)

[**Funciones complementarias** 62](#_Toc50970061)

[**Funciones comunes** 63](#_Toc50970062)

[**RAZAS** 65](#_Toc50970063)

[**Funciones comunes** 66](#_Toc50970064)

[**VETERINARIOS** 68](#_Toc50970065)

[**registrarColegiaturaPersona** 68](#_Toc50970066)

[**cancelarPerfilPersona** 69](#_Toc50970067)

[**cargarDatosIniciales** 71](#_Toc50970068)

[**Funciones comunes** 71](#_Toc50970069)

[**Otros posibles contratos inteligentes** 73](#_Toc50970070)

[**TITULOS** 73](#_Toc50970071)

[solicitarTitulo 73](#_Toc50970072)

[obtenerTitulo 73](#_Toc50970073)

[consultarTitulos 73](#_Toc50970074)

[**EXPOSICIONES Y CONCURSOS** 74](#_Toc50970075)

[registarShow 74](#_Toc50970076)

[registarResultadoShow 74](#_Toc50970077)

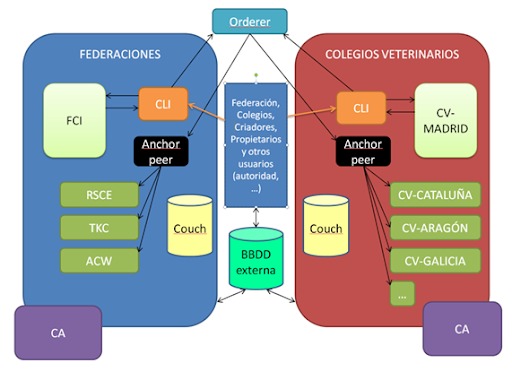
[cagarResultadosShow 74](#_Toc50970078)

[consultarResultadosShow 74](#_Toc50970079)

[**ADN** 75](#_Toc50970080)

[**ENFERMEDADES / TRATAMIENTOS** 75](#_Toc50970081)

# **Contratos divididos por Organizaciones**



Los contratos que utiliza cada organización son los siguientes

## **Federaciones Caninas**

* Personas
* Perfil
* Perros
* Afijos
* Solicitudes
* Veterinarios
* Microchip
* Razas
* Exposiciones y concursos
* Títulos

## **Colegios de Veterinarios**

* Personas
* Perfil
* Perros
* Veterinarios
* Vacunas
* Microchips
* Razas

# **Contratos inteligentes**

## **Parámetros de entrada/salida de las funciones**

Las funciones definidas en los contratos disponen de parameros de entrada específicos para su funcionamiento.

Como regla general dispones 2 argumentos:

args[0] : de tipo JSON con datos específicos de a función

args[1]: de tipo JSON con los datos de seguridad donde se identifica al usuario que invoca la función.

Por el contrario, todas las funciones definidas en los contratos tendrán un parámetro de salida de tipo Response:

type Response struct {

// A status code that should follow the HTTP status codes.

Status [int32](https://godoc.org/builtin#int32) `protobuf:"varint,1,opt,name=status,proto3" json:"status,omitempty"`

// A message associated with the response code.

Message [string](https://godoc.org/builtin#string) `protobuf:"bytes,2,opt,name=message,proto3" json:"message,omitempty"`

// A payload that can be used to include metadata with this response.

Payload [][byte](https://godoc.org/builtin#byte) `protobuf:"bytes,3,opt,name=payload,proto3" json:"payload,omitempty"`

XXX\_NoUnkeyedLiteral struct{} `json:"-"`

XXX\_unrecognized [][byte](https://godoc.org/builtin#byte) `json:"-"`

XXX\_sizecache [int32](https://godoc.org/builtin#int32) `json:"-"`

}

Un perro corriendo en el pasto

Descripción generada automáticamente

## **Funciones comunes de los contratos**

Existen un conjunto defunciones que se definen dentro de los contratos:

### **ejecutarConsulta**

Permite realizar consultas al sistema a través del lenguaje de consulta de Mango, que se expresa como un objeto JSON que describe los filtros de los documentos de interés que se desean consultar.

### **asignarEstado**

Asigna un valor de estado al sistema.

### **borrarEstado**

Elimina un valor de estado del sistema.

### **consultarEstado**

Consulta un valor de estado del sistema.

### **consultarRangoEstados**

Consulta los valores de estado del sistema comprendidos entre dos valores.

### **getQueryResultForQueryString**

Devuelve en el siguiente formato JSON el resultado de una consulta, donde el TipoEstado tendrá la definición especifica del estado consultado.

type TipoQueryEstado struct {

    Key    string             `json:"Key"`

    Record TipoEstado `json:"Record"`

}

**Imagen que contiene perro, parado, trineo, viendo

Descripción generada automáticamente**

## **PERSONAS**

Este contrato inteligente es el encargado de gestionar los datos de las personas que intervienen en las solicitudes de los usuarios del sistema.

Actualmente gestiona los datos identificativos de:

* Criadores (propietarios de afijos)
* Propietarios de Perros
* Veterinarios
* Miembros de Federaciones

El presente contrato inteligente esta diseñado para admitir en el futuro ampliaciones de la blockchain, dando la posibilidad de albergar los datos de identificación de otro tipo de usuarios o datos asociados.

Por ejemplo, incorporar:

* Jueces de certámenes caninos
* Miembros de Asociaciones caninas
* Miembros de Club de raza caninas y de grupos
* Miembros de laboratorios (para registros de ADN)
* …



Funciones que incorpora el contrato inteligente:

### **registrarPersona**

Introduce en el sistema los datos necesarios para la identificación de una nueva persona.

* **Argumentos entrada:**

type registrarPersonas struct {

Nombre                 string `json:"Nombre"`

    Apellido1              string `json:"Apellido1"`

    Apellido2              string `json:"Apellido2"`

    TipoDocumento          string `json:"TipoDocumento"`

    IdentificadorDocumento string `json:"IdentificadorDocumento"`

    PaisEmisor             string `json:"PaisEmisor"`

    Certificado            string `json:"Certificado"`

}

|  |  |
| --- | --- |
| El contrato realiza las siguientes validaciones:   * El número de argumentos de entrada sea 1 * El formato JSON de args[0] sea valido * Los argumentos Nombre, Apellido1, Apellido2, TipoDocumento, IdentificadorDocumento y PaisEmisor tenga un valor. * No exista otra persona registrada con eses mismo TipoDocumento, IdentificadorDocumento y PaisEmisor. |  |

* **Argumentos salida:**

Devuelve el registro insertado:

type Personas struct {

ObjectType string `json:"docType"`

IDPersona int `json:"IDPersona"`

Nombre string `json:"Nombre"`

Apellido1 string `json:"Apellido1"`

Apellido2 string `json:"Apellido2"`

TipoDocumento string `json:"TipoDocumento"`

IdentificadorDocumento string `json:"IdentificadorDocumento"`

PaisEmisor string `json:"PaisEmisor"`

Certificado string `json:"Certificado"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

### **registrarDocumentoIdentidad**

Introduce en el sistema los datos otros documentos de identidad de la persona, válidos para identificar a una persona registra.

Por ejemplo: las mujeres rumanas cuando se casa cambian el documento de identidad por el de su marido añadiendo un digito adicional o en España cuando un persona adquiere la nacionalidad se le cambia el NIE por el NIF.

### **modificarNombreApellidos**

Modifica en el sistema el nombre y/o apellido de una persona registra.

Por ejemplo: en caso de cambio de nombre como Ignacio por Iñaki o cambio de orden de los apellidos.

Imagen que contiene café, perro, pasto, persona

Descripción generada automáticamente

**cargarDatosIniciales**

Registrar en el sistema los datos de las propietarios, veterinarios y personal de las federaciones definidos en un fichero de texto con el siguiente formato JSON:

type Personas struct {

ObjectType string `json:"docType"`

IDPersona int `json:"IDPersona"`

Nombre string `json:"Nombre"`

Apellido1 string `json:"Apellido1"`

Apellido2 string `json:"Apellido2"`

TipoDocumento string `json:"TipoDocumento"`

IdentificadorDocumento string `json:"IdentificadorDocumento"`

PaisEmisor string `json:"PaisEmisor"`

Certificado string `json:"Certificado"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

* **Argumentos entrada:**

NombreFichero string

* **Argumentos salida:**

Devuelve el número de registros insertados:

NumeroRegistros string

### **Funciones comunes**

Funciones de los contratos del sistema incorporadas a este contrato:

* **getQueryResultForQueryString**
* **asignarEstado**
* **borrarEstado**
* **consultarEstado**
* **consultarRangoEstados**
* **ejecutarConsulta**



**Plan de pruebas**

A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

export CHANNEL\_NAME=netcanchannel

export CA\_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-cert.pem

export ORDENER\_URL=orderer.netcan.com:7050

**# PERSONAS**

peer chaincode invoke -n personas -c '{"function":"registrarPersona","Args":["{\"Nombre\":\"Unai\",\"Apellido1\":\"Ares\",\"Apellido2\":\"Icaran\",\"TipoDocumento\":\"NIF\",\"IdentificadorDocumento\":\"30624315E\",\"PaisEmisor\":\"Spain\"}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# ERRORES PERSONAS**

peer chaincode invoke -n personas -c '{"function":"registrarPersona","Args":[""]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n personas -c '{"function":"registrarPersona","Args":["{Nombre\":\"Unai\",\"Apellido1\":\"Ares\",\"Apellido2\":\"Icaran\",\"TipoDocumento\":\"NIF\",\"IdentificadorDocumento\":\"30624315E\",\"PaisEmisor\":\"Spain\"}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n personas -c '{"function":"registrarPersona","Args":["{\"Nombre\":\"Unai\",\"Apellido1\":\"Ares\",\"Apellido2\":\"Icaran\",\"TipoDocumento\":\"NIF\",\"IdentificadorDocumento\":\"30624315E\",\"PaisEmisor\":\"Spain\"}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n personas -c '{"function":"registrarPersona","Args":["{\"Apellido1\":\"Ares\",\"Apellido2\":\"Icaran\",\"TipoDocumento\":\"NIF\",\"IdentificadorDocumento\":\"30624315E\",\"PaisEmisor\":\"Spain\"}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# CARGAS INICIALES**

peer chaincode invoke -n personas -c '{"function":"cargarDatosIniciales","Args":["./json/personas\_001.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

sleep 5

peer chaincode invoke -n personas -c '{"function":"asignarEstado","Args":["PERSONAS", "{\"docType\":\"CONTADOR\",\"IDMaximo\":1000}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

Imagen que contiene perro, interior, persona, café

Descripción generada automáticamente

## **AFIJOS**

Este contrato inteligente es el encargado de gestionar los datos de los afijos de los criadores del sistema.

Un **afijo de un criador** es una denominación o marca personal registrado a través de una sociedad o federación canina que le autoriza a su titular o titulares a utilizarlo en la inscripción de camadas.



Funciones que incorpora el contrato inteligente:

### **registrarAfijo**

Introduce en el sistema los datos necesarios para registrar y autorizar el uso de un nuevo Afijo de Criador por parte de uno o más usuarios cumpliendo los siguientes requisitos:

* Ningún solicitante puede ser propietario o copropietario de otro afijo
* El nombre afijo no debe existir
* **Argumentos entrada:**

type tipoRegistrarAfijoPropietario struct {

IDPersona int `json:"IDPersona"`

}

type tipoRegistrarAfijo struct {

Nombre string `json:"Nombre"`

Propietarios []tipoRegistrarAfijoPropietario `json:"Propietarios"`

}

type TipoSeguridad struct {

    IDPersona int `json:"IDPersona"`

}

|  |  |
| --- | --- |
| El contrato realiza las siguientes validaciones:   * El número de argumentos de entrada sea 2 * El formato JSON de args[0] y args[1] sea valido * Los argumentos Nombre y Propietarios tenga un valor. * No existe otro Afijo con el mismo nombre * Los propietarios están registrados * Alguno de los propietarios no tiene actualmente otro afijo asignado como propietario o copropietario * La persona que invoca la acción esta registrada y dispone de los permisos para realizar dicha acción |  |

* **Argumentos salida:**

Devuelve los registros insertados:

type Afijos struct {

ObjectType string `json:"docType"`

IDAfijo int `json:"IDAfijo"`

Nombre string `json:"Nombre"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

type AfijosPropietarios struct {

ObjectType string `json:"docType"`

IDAfijoPropietario int `json:"IDAfijoPropietario"`

IDAfijo int `json:"IDAfijo"`

IDPersona int `json:"IDPersona"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

### **registrarCambioPropietario**

Registrar en el sistema los datos de los nuevos propietarios de un afijo, dando de baja el registro de los actuales propietarios.

* **Argumentos entrada:**

type tiporegistrarCancelacionAfijo struct {

        IDAfijo int `json:"IDAfijo"`

    }

type TipoSeguridad struct {

    IDPersona int `json:"IDPersona"`

}

|  |  |
| --- | --- |
| El contrato realiza las siguientes validaciones:   * El número de argumentos de entrada sea 2 * El formato JSON de args[0] y args[1] sea valido * El IDAfijo tenga un valor y corresponda a un afijo activo. * Los nuevos propietarios están registrados y no tiene ninguno de ellos otro afijo asignado como propietario o copropietario * La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción |  |



* **Argumentos salida:**

Devuelve los registros insertados:

type Afijos struct {

ObjectType string `json:"docType"`

IDAfijo int `json:"IDAfijo"`

Nombre string `json:"Nombre"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

type AfijosPropietariosBaja struct {

ObjectType string `json:"docType"`

IDAfijoPropietario int `json:"IDAfijoPropietario"`

IDAfijo int `json:"IDAfijo"`

IDPersona int `json:"IDPersona"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

type AfijosPropietariosAlta struct {

ObjectType string `json:"docType"`

IDAfijoPropietario int `json:"IDAfijoPropietario"`

IDAfijo int `json:"IDAfijo"`

IDPersona int `json:"IDPersona"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

**registrarCancelacionAfijo**

Registrar en el sistema la cancelación del Afijo de Criador, cuyo nombre no podrá ser utilizado por ninguna otra persona.

* **Argumentos entrada:**

type tiporegistrarCancelacionAfijo struct {

    IDAfijo int `json:"IDAfijo"`

}

type TipoSeguridad struct {

IDPersona int `json:"IDPersona"`

}



|  |  |
| --- | --- |
| El contrato realiza las siguientes validaciones:   * El número de argumentos de entrada sea 2 * El formato JSON de args[0] y args[1] sea valido * El IDAfijo tenga un valor y corresponda a un afijo activo. * La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción |  |

* **Argumentos salida:**

Devuelve los registros insertados:

type Afijos struct {

ObjectType string `json:"docType"`

IDAfijo int `json:"IDAfijo"`

Nombre string `json:"Nombre"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

type AfijosPropietarios struct {

ObjectType string `json:"docType"`

IDAfijoPropietario int `json:"IDAfijoPropietario"`

IDAfijo int `json:"IDAfijo"`

IDPersona int `json:"IDPersona"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

**cargarDatosIniciales**

Insertar en el sistema los datos los distintos Afijos registrados por las federaciones en un fichero de texto con el siguiente formato JSON:

type Afijos struct {

ObjectType string `json:"docType"`

IDAfijo int `json:"IDAfijo"`

Nombre string `json:"Nombre"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

* **Argumentos entrada:**

NombreFichero string

* **Argumentos salida:**

Devuelve el número de registros insertados:

NumeroRegistros string

### **cargarDatosIniciales\_Propietarios**

Insertar en el sistema los datos los Propietarios de los distintos Afijos registrados por las federaciones en un fichero de texto con el siguiente formato JSON:

type AfijosPropietarios struct {

ObjectType string `json:"docType"`

IDAfijoPropietario int `json:"IDAfijoPropietario"`

IDAfijo int `json:"IDAfijo"`

IDPersona int `json:"IDPersona"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

* **Argumentos entrada:**

NombreFichero string

* **Argumentos salida:**

Devuelve el número de registros insertados:

NumeroRegistros string

### **consultarDatosAfijo**

Devuelve al solicitante los datos que sean públicos del afijo consultado.

### **obtenerCertificadoAfijo**

Proporciona al propietario un fichero que certifica la propiedad de este.

### **Funciones comunes**

Funciones de los contratos del sistema incorporadas a este contrato:

* **getQueryResultForQueryString**
* **asignarEstado**
* **borrarEstado**
* **consultarEstado**
* **consultarRangoEstados**
* **ejecutarConsulta**



**Plan de pruebas**

A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

export CHANNEL\_NAME=netcanchannel

export CA\_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-cert.pem

export ORDENER\_URL=orderer.netcan.com:7050

**# AFIJOS**

peer chaincode invoke -n afijos -c '{"function":"registrarAfijo","Args":["{\"Nombre\":\"NUEVO AFIJO\",\"Propietarios\":[{\"IDPersona\":700},{\"IDPersona\":701},{\"IDPersona\":702}]}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n afijos -c '{"function":"registrarCambioPropietario","Args":["{\"IDAfijo\":51,\"Propietarios\":[{\"IDPersona\":800},{\"IDPersona\":801},{\"IDPersona\":802},{\"IDPersona\":803}]}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n afijos -c '{"function":"registrarCancelacionAfijo","Args":["{\"IDAfijo\":51}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# ERRORES AFIJOS**

peer chaincode invoke -n afijos -c '{"function":"registrarAfijo","Args":["{\"Nombre\":\"NUEVO AFIJO\",\"Propietarios\":[{\"IDPersona\":700},{\"IDPersona\":701},{\"IDPersona\":702}]}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n afijos -c '{"function":"registrarAfijo","Args":["{\"Nombre\":\"OTRO AFIJO\",\"Propietarios\":[{\"IDPersona\":700},{\"IDPersona\":801},{\"IDPersona\":802}]}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n afijos -c '{"function":"registrarCambioPropietario","Args":["{\"IDAfijo\":51,\"Propietarios\":[{\"IDPersona\":1},{\"IDPersona\":2},{\"IDPersona\":3},{\"IDPersona\":4}]}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n afijos -c '{"function":"registrarCambioPropietario","Args":["{\"IDAfijo\":1051,\"Propietarios\":[{\"IDPersona\":800},{\"IDPersona\":801},{\"IDPersona\":802},{\"IDPersona\":803}]}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n afijos -c '{"function":"registrarCambioPropietario","Args":["{\"IDAfijo\":51,\"Propietarios\":[{\"IDPersona\":88888888888888},{\"IDPersona\":2},{\"IDPersona\":3},{\"IDPersona\":4}]}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n afijos -c '{"function":"registrarCancelacionAfijo","Args":["{\"IDAfijo\":51}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n afijos -c '{"function":"registrarCambioPropietario","Args":["{\"IDAfijo\":555555551,\"Propietarios\":[{\"IDPersona\":800},{\"IDPersona\":801},{\"IDPersona\":802},{\"IDPersona\":803}]}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# CARGAS INICIALES**

peer chaincode invoke -n $CC\_NOMBRE\_AFIJOS -c '{"function":"cargarDatosIniciales","Args":["./json/afijos.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n $CC\_NOMBRE\_AFIJOS -c '{"function":"asignarEstado","Args":["AFIJOS", "{\"docType\":\"CONTADOR\",\"IDMaximo\":50}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n $CC\_NOMBRE\_AFIJOS -c '{"function":"cargarDatosIniciales\_Propietarios","Args":["./json/afijos\_propietarios.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

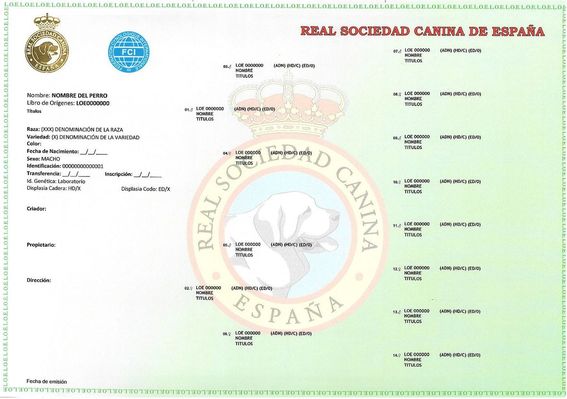
peer chaincode invoke -n $CC\_NOMBRE\_AFIJOS -c '{"function":"asignarEstado","Args":["AFIJOS\_PROPIETARIOS", "{\"docType\":\"CONTADOR\",\"IDMaximo\":50}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

Un perro corriendo en el pasto

Descripción generada automáticamente

## **PERROS**

Este contrato inteligente es el encargado de gestionar el registro de ejemplares canino, identificando su origen, pureza de raza, mencionando sus ascendentes y descendientes.



Permite registrar a los propietarios de ejemplar hembra registrar el nacimiento de una nueva camada en un periodo no superior a 1 mes desde su nacimiento (identificando a los progenitores yel nombre y sexo de cada cachorro

También permite el registro de propiedad de perros mestizos en los cuales algún de los progenitores es desconocido o es un mestizo, y los cambios de propietarios, así como la defunción de los ejemplares.

Funciones que incorpora el contrato inteligente:

### **registrarPerro**

Introduce en el sistema los datos necesarios para identificar el origen genealógico de uno o varios perros y registrar sus propietarios.

Permite registrar una camada o un ejemplar, determinando la pureza de raza y el afijo de criador según los criterios de validación detallados más adelante.

* **Argumentos entrada:**

type tipoRegistrarCamadaPerro struct {

Nombre string `json:"Nombre"`

IDSexo int `json:"IDSexo"`

}

type tipoRegistrarCamadaPropietario struct {

IDPersona int `json:"IDPersona"`

}

type tipoRegistrarCamada struct {

Perros []tipoRegistrarCamadaPerro `json:"Perros"`

IDPerroMadre int `json:"IDPerroMadre"`

IDPerroPadre int `json:"IDPerroPadre"`

IDAfijo int `json:"IDAfijo"`

IDRaza int `json:"IDRaza"`

FechaNacimiento string `json:"FechaNacimiento"`

Propietarios []tipoRegistrarCamadaPropietario `json:"Propietarios"`

}

type TipoSeguridad struct {

    IDPersona int `json:"IDPersona"`

}

|  |  |
| --- | --- |
| El contrato realiza las siguientes validaciones:   * El número de argumentos de entrada sea 2 * El formato JSON de args[0] y args[1] sea valido * En caso de que la madre no sea desconocida (IDMadre≠0):   + El IDMadre está registrado y sigue viva.   + El IDMadre este registrada como ejemplar Hembra (IDSexo = 0)   + Su edad esta comprendida entre 1 y 10 años (sino perderá la pureza de raza y se asignará IDRaza = 0)   + Si todos los propietarios de la Madre son propietarios del mismo Afijo de Criador (sino perderá la denominación del afijo, y se asignara IDAfijo = 0) que serán los propietarios del ejemplar * En caso de que la madre sea desconocida (IDMadre = 0):   + Los nuevos propietarios están registrados * En caso de que el padra no sea desconocida (IDPadre≠0):   + El IDPadre está registrado y sigue vivo.   + El IDPadre este registrado como ejemplar Macho (IDSexo = 1)   + Su edad está comprendida entre 1 y 12 años (sino perderá la pureza de raza y se asignará IDRaza = 0) * Si la raza de los dos progenitores es igual (sino perderá la pureza de raza y se asignará IDRaza=0) * Los argumentos Nombre e IDSexo tenga un valor valido:   + En el caso de que IDAfijo ≠ 0 no exista para ese afijo de criador otro ejemplar con el mismo nombre   + IDSexo contiene o 0 (HEMBRA) 0 1 (MACHO) * El argumento FechaNacimiento tenga un valor q los últimos 30 días (sino perderá la pureza de raza y se asignará IDRaza = 0) * La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción |  |

Mujer sonriendo con perro al lado

Descripción generada automáticamente

* **Argumentos salida:**

Devuelve los registros insertados:

type Perros struct {

ObjectType string `json:"docType"`

IDPerro int `json:"IDPerro"`

Nombre string `json:"Nombre"`

IDAfijo int `json:"IDAfijo"`

IDSexo int `json:"IDSexo"`

IDPerroMadre int `json:"IDPerroMadre"`

IDPerroPadre int `json:"IDPerroPadre"`

IDRaza int `json:"IDRaza"`

FechaNacimiento string `json:"FechaNacimiento"`

FechaDefuncion string `json:"FechaDefuncion"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

type PerrosPropietarios struct {

ObjectType string `json:"docType"`

IDPerroPropietario int `json:"IDPerroPropietario"`

IDPerro int `json:"IDPerro"`

IDPersona int `json:"IDPersona"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

### **registrarCambioPropietario**

Registrar en el sistema los datos de los nuevos propietarios de un ejemplar, dando de baja el registro de los actuales propietarios.

* **Argumentos entrada:**

type TipoRegistrarCambioPropietarioPropietario struct {

IDPersona int `json:"IDPersona"`

}

type TipoRegistrarCambioPropietario struct {

IDPerro int `json:"IDPerro"`

Propietarios []TipoRegistrarCambioPropietarioPropietario `json:"Propietarios"`

}

type TipoSeguridad struct {

    IDPersona int `json:"IDPersona"`

}

|  |  |
| --- | --- |
| El contrato realiza las siguientes validaciones:   * El número de argumentos de entrada sea 2 * El formato JSON de args[0] y args[1] sea valido * El IDAfijo tenga un valor y corresponda a un afijo activo. * Los nuevos propietarios están registrados * Alguno de los nuevos propietarios no tiene actualmente otro afijo asignado como propietario o copropietario * La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción |  |

Imagen que contiene perro, animal, mamífero, exterior

Descripción generada automáticamente

* **Argumentos salida:**

Devuelve los registros insertados:

type Perros struct {

ObjectType string `json:"docType"`

IDPerro int `json:"IDPerro"`

Nombre string `json:"Nombre"`

IDAfijo int `json:"IDAfijo"`

IDSexo int `json:"IDSexo"`

IDPerroMadre int `json:"IDPerroMadre"`

IDPerroPadre int `json:"IDPerroPadre"`

IDRaza int `json:"IDRaza"`

FechaNacimiento string `json:"FechaNacimiento"`

FechaDefuncion string `json:"FechaDefuncion"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

type PerrosPropietariosBaja struct {

ObjectType string `json:"docType"`

IDPerroPropietario int `json:"IDPerroPropietario"`

IDPerro int `json:"IDPerro"`

IDPersona int `json:"IDPersona"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

type PerrosPropietariosAlta struct {

ObjectType string `json:"docType"`

IDPerroPropietario int `json:"IDPerroPropietario"`

IDPerro int `json:"IDPerro"`

IDPersona int `json:"IDPersona"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

### **registrarDefuncionPerro**

Registrar en el sistema la baja de un ejemplar por defunción.

* **Argumentos entrada:**

type tiporegistrarCancelacionAfijo struct {

        IDAfijo int `json:"IDAfijo"`

}

type TipoSeguridad struct {

IDPersona int `json:"IDPersona"`

}

|  |  |
| --- | --- |
| El contrato realiza las siguientes validaciones:   * El número de argumentos de entrada sea 2 * El formato JSON de args[0] y args[1] sea valido * El IDPerro tenga un valor y corresponda a un ejemplar vivo. * La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción |  |

* **Argumentos salida:**

Devuelve los registros insertados:

type Perros struct {

ObjectType string `json:"docType"`

IDPerro int `json:"IDPerro"`

Nombre string `json:"Nombre"`

IDAfijo int `json:"IDAfijo"`

IDSexo int `json:"IDSexo"`

IDPerroMadre int `json:"IDPerroMadre"`

IDPerroPadre int `json:"IDPerroPadre"`

IDRaza int `json:"IDRaza"`

FechaNacimiento string `json:"FechaNacimiento"`

FechaDefuncion string `json:"FechaDefuncion"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

type PerrosPropietarios struct {

ObjectType string `json:"docType"`

IDPerroPropietario int `json:"IDPerroPropietario"`

IDPerro int `json:"IDPerro"`

IDPersona int `json:"IDPersona"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

**Imagen que contiene perro, animal, exterior, mamífero

Descripción generada automáticamente**

### **registrarReconocimientoRaza**

Registrar en el sistema la certificación de la Pueeza de raza de un ejemplar cuyos ascendentes no están determinados o registrados en un libro de origen genealógico.

Este hecho se realiza de manera excepcional por una de las siguientes causas:

* Un juez especialista de Raza certifica en un evento organizado por las federaciones que el ejemplar cumple con todas las características del estándar de la raza.
* El propietario presenta una Certificado de un Libro de Origen Genealógico reconocido y la federación verifica la autenticación del documento con el organismo que gestiona el libro.
* **Argumentos entrada:**

type PerroReconocimientoRaza struct {

IDPerro int `json:"IDPerro"`

IDRaza int `json:"IDRaza"`

Justificacion string `json:"Justificacion"`

}

type TipoSeguridad struct {

IDPersona int `json:"IDPersona"`

}

|  |  |
| --- | --- |
| El contrato realiza las siguientes validaciones:   * El número de argumentos de entrada sea 2 * El formato JSON de args[0] y args[1] sea valido * El IDPerro tenga un valor y corresponda a un ejemplar vivo. * El IDRaza tenga un valor y corresponda a un afijo activo. * La Justificación tenga un valor * La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción |  |

* **Argumentos salida:**

Devuelve los registros insertados:

type Perros struct {

ObjectType string `json:"docType"`

IDPerro int `json:"IDPerro"`

Nombre string `json:"Nombre"`

IDAfijo int `json:"IDAfijo"`

IDSexo int `json:"IDSexo"`

IDPerroMadre int `json:"IDPerroMadre"`

IDPerroPadre int `json:"IDPerroPadre"`

IDRaza int `json:"IDRaza"`

FechaNacimiento string `json:"FechaNacimiento"`

FechaDefuncion string `json:"FechaDefuncion"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

type PerrosPropietarios struct {

ObjectType string `json:"docType"`

IDPerroPropietario int `json:"IDPerroPropietario"`

IDPerro int `json:"IDPerro"`

IDPersona int `json:"IDPersona"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}



### **cargarDatosIniciales**

Registrar en el sistema los ejemplares definidos en un fichero de texto con el siguiente formato JSON:

type Perros struct {

ObjectType string `json:"docType"`

IDPerro int `json:"IDPerro"`

Nombre string `json:"Nombre"`

IDAfijo int `json:"IDAfijo"`

IDSexo int `json:"IDSexo"`

IDPerroMadre int `json:"IDPerroMadre"`

IDPerroPadre int `json:"IDPerroPadre"`

IDRaza int `json:"IDRaza"`

FechaNacimiento string `json:"FechaNacimiento"`

FechaDefuncion string `json:"FechaDefuncion"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

* **Argumentos entrada:**

NombreFichero string

* **Argumentos salida:**

Devuelve el número de registros insertados:

NumeroRegistros string

### **cargarDatosIniciales\_Propietarios**

Registrar en el sistema los propietarios de los ejemplares definidos en un fichero de texto con el siguiente formato JSON:

type PerrosPropietarios struct {

ObjectType string `json:"docType"`

IDPerroPropietario int `json:"IDPerroPropietario"`

IDPerro int `json:"IDPerro"`

IDPersona int `json:"IDPersona"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

* **Argumentos entrada:**

NombreFichero string

* **Argumentos salida:**

Devuelve el número de registros insertados:

NumeroRegistros string

### **consultarDatosEjemplar**

Consultar los datos del ejemplar. La información presentada variará dependiendo de la persona que realiza la consulta.

### **obtenerCertificadoRegistro**

Obtener un certificado con los datos del registro que certifica su inscripción en el libro genealógico.

### **obtenerPedigri**

Obtener un fichero digital que contiene los datos del ejemplar y su línea genealógica que certifica su pureza de raza.

### **registrarCesionTemporal**

Solicitar el registro de cesión temporal de la propiedad a uno o varios criadores por un tiempo limitado.

### **Funciones comunes**

Funciones de los contratos del sistema incorporadas a este contrato:

* **getQueryResultForQueryString**
* **asignarEstado**
* **borrarEstado**
* **consultarEstado**
* **consultarRangoEstados**
* **ejecutarConsulta**



**Plan de pruebas**

A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

export CHANNEL\_NAME=netcanchannel

export CA\_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-cert.pem

export ORDENER\_URL=orderer.netcan.com:7050

**# PERROS**

peer chaincode invoke -n perros -c '{"function":"registrarPerro","Args":["{\"Perros\":[{\"Nombre\":\"PERRO\_0001\",\"IDSexo\":1},{\"Nombre\":\"PERRA\_0001\",\"IDSexo\":0}],\"IDPerroMadre\":181,\"IDPerroPadre\":193,\"FechaNacimiento\":\"2020-08-23\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n perros -c '{"function":"registrarPerro","Args":["{\"Perros\":[{\"Nombre\":\"PERRO\_0002\",\"IDSexo\":1},{\"Nombre\":\"PERRA\_0002\",\"IDSexo\":0}],\"IDPerroMadre\":0,\"IDPerroPadre\":0,\"FechaNacimiento\":\"2020-08-23\",\"Propietarios\":[{\"IDPersona\":600},{\"IDPersona\":601}]}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n perros -c '{"function":"registrarPerro","Args":["{\"Perros\":[{\"Nombre\":\"PERRO\_0001\",\"IDSexo\":1},{\"Nombre\":\"PERRA\_0001\",\"IDSexo\":0}],\"IDPerroMadre\":0,\"IDPerroPadre\":193,\"FechaNacimiento\":\"2020-08-23\",\"Propietarios\":[{\"IDPersona\":600},{\"IDPersona\":601}]}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n $CC\_NOMBRE\_PERROS -c '{"function":"registrarCambioPropietario","Args":["{\"IDPerro\":5760,\"Propietarios\":[{\"IDPersona\":401},{\"IDPersona\":402},{\"IDPersona\":403},{\"IDPersona\":404}]}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n $CC\_NOMBRE\_PERROS -c '{"function":"registrarDefuncionPerro","Args":["{\"IDPerro\":100,\"FechaDefuncion\":\"2020-08-08\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n perros -c '{"function":"registrarReconocimientoRaza","Args":["{\"IDPerro\":37,\"IDRaza\":126,\"Justificacion\":\"Ha sido reconocida su raza por el Juez PEPE GUTIERREZ DOSSANTOS\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# ERRORES PERROS**

peer chaincode invoke -n perros -c '{"function":"registrarPerro","Args":["{\"Perros\":[{\"Nombre\":\"PERRO\_0001\",\"IDSexo\":1},{\"Nombre\":\"PERRA\_0001\",\"IDSexo\":0}],\"IDPerroMadre\":181,\"IDPerroPadre\":3,\"FechaNacimiento\":\"2020-08-23\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n perros -c '{"function":"registrarPerro","Args":["{\"Perros\":[{\"Nombre\":\"PERRO\_0001\",\"IDSexo\":1},{\"Nombre\":\"PERRA\_0001\",\"IDSexo\":0}],\"IDPerroMadre\":181,\"IDPerroPadre\":10,\"FechaNacimiento\":\"2020-08-23\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n perros -c '{"function":"registrarPerro","Args":["{\"Perros\":[{\"Nombre\":\"PERRO\_0001\",\"IDSexo\":1},{\"Nombre\":\"PERRA\_0001\",\"IDSexo\":0}],\"IDPerroMadre\":0,\"IDPerroPadre\":193,\"FechaNacimiento\":\"2020-08-23\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n $CC\_NOMBRE\_PERROS -c '{"function":"registrarCambioPropietario","Args":["{\"IDPerro\":577777777760,\"Propietarios\":[{\"IDPersona\":401},{\"IDPersona\":402},{\"IDPersona\":403},{\"IDPersona\":404}]}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n $CC\_NOMBRE\_PERROS -c '{"function":"registrarCambioPropietario","Args":["{\"IDPerro\":5760,\"Propietarios\":[{\"IDPersona\":401},{\"IDPersona\":402},{\"IDPersona\":444444444403},{\"IDPersona\":404}]}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME



Peer chaincode invoke -n $CC\_NOMBRE\_PERROS -c '{"function":"registrarDefuncionPerro","Args":["{\"IDPerro\":100,\"FechaDefuncion\":\"2020-08-08\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n $CC\_NOMBRE\_PERROS -c '{"function":"registrarDefuncionPerro","Args":["{\"IDPerro\":1000000000,\"FechaDefuncion\":\"2020-08-08\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n $CC\_NOMBRE\_PERROS -c '{"function":"registrarDefuncionPerro","Args":["{\"IDPerro\":100\"FechaDefuncion\":\"2020-08-08\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n perros -c '{"function":"registrarReconocimientoRaza","Args":["{\"IDPerro\":12,\"IDRaza\":126,\"Justificacion\":\"Ha sido reconocida su raza por el Juez PEPE GUTIERREZ DOSSANTOS\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# CARGAS INICIALES**

peer chaincode invoke -n perros -c '{"function":"cargarDatosIniciales","Args":["./json/perros\_001.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

sleep 7s

peer chaincode invoke -n perros -c '{"function":"cargarDatosIniciales","Args":["./json/perros\_002.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

sleep 7s

peer chaincode invoke -n perros -c '{"function":"cargarDatosIniciales","Args":["./json/perros\_003.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

sleep 7s

peer chaincode invoke -n perros -c '{"function":"cargarDatosIniciales","Args":["./json/perros\_004.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

sleep 7s

peer chaincode invoke -n perros -c '{"function":"cargarDatosIniciales","Args":["./json/perros\_005.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

sleep 7s

peer chaincode invoke -n perros -c '{"function":"asignarEstado","Args":["PERROS", "{\"docType\":\"CONTADOR\",\"IDMaximo\":5759}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n perros -c '{"function":"cargarDatosIniciales\_Propietarios","Args":["./json/perros\_propietarios\_001.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

sleep 7s

peer chaincode invoke -n perros -c '{"function":"cargarDatosIniciales\_Propietarios","Args":["./json/perros\_propietarios\_002.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

sleep 7s

peer chaincode invoke -n perros -c '{"function":"cargarDatosIniciales\_Propietarios","Args":["./json/perros\_propietarios\_003.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

sleep 7s

peer chaincode invoke -n perros -c '{"function":"cargarDatosIniciales\_Propietarios","Args":["./json/perros\_propietarios\_004.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

sleep 7s

peer chaincode invoke -n perros -c '{"function":"cargarDatosIniciales\_Propietarios","Args":["./json/perros\_propietarios\_005.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

sleep 7s

peer chaincode invoke -n perros -c '{"function":"asignarEstado","Args":["PERROS\_PROPIETARIOS", "{\"docType\":\"CONTADOR\",\"IDMaximo\":5759}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME



## **SOLICITUDES**

Este contrato inteligente es el encargado de gestionar las solicitudes de los usuarios cuando es necesario realizar la validación o autorización de la acción por más de una persona.

Por ejemplo:

* Cuando se desea realizar un cambio de propietario de un ejemplar y este pertenece a varias personas, requiere la autorización de ambas personas.
* Cuando se desea registrar el nacimiento de una nueva camada y los progenitores pertenecen a dos criadores o propietarios distintos, no basta con que se registre la solicitud por parte de uno de ellos sino de ambos.



La solicitud es registrada en el sistema por cualquiera de los usuarios implicados, quedando pendiente su ejecución hasta que todos los implicados validan o rechazan el contenido de la solicitud o caduca por transcurrir un tiempo desde su registro.

Una vez que todos los implicados han validado la solicitud esta se ejecutara automáticamente.

En el caso en el que solo sea necesaria la intervención de un usuario y sea este mismo usuario el que registrara la solicitud, esta se ejecutar a también automáticamente.

Con estas certificaciones se evitan fraudes que son comunes, como por ejemplo vender un perro a un nuevo propietario falsificando su origen genealógico (por ejemplo, es hijo del campeón del España)

Las funciones que incorpora el contrato inteligente son similares a las vistas anteriormente en otros contratos:

* **Argumentos entrada:**

Serán los mismos parámetros de entrada que tendría la llamada a la función si realizara la acción directamente.

Por ejemplo:

**solicitarRegistrarPerro** tendrá los mismos parámetros de entrada la función **registrarPerro** del contrato **PERROS:**

type tipoRegistrarCamadaPerro struct {

Nombre string `json:"Nombre"`

IDSexo int `json:"IDSexo"`

}

type tipoRegistrarCamadaPropietario struct {

IDPersona int `json:"IDPersona"`

}

type tipoRegistrarCamada struct {

Perros []tipoRegistrarCamadaPerro `json:"Perros"`

IDPerroMadre int `json:"IDPerroMadre"`

IDPerroPadre int `json:"IDPerroPadre"`

IDAfijo int `json:"IDAfijo"`

IDRaza int `json:"IDRaza"`

FechaNacimiento string `json:"FechaNacimiento"`

Propietarios []tipoRegistrarCamadaPropietario `json:"Propietarios"`

}

type TipoSeguridad struct {

    IDPersona int `json:"IDPersona"`

}

* **Argumentos salida:**

Si la solicitud se puede ejecutar automáticamente devolverá los mismos parámetros de salida que la función **registrarPerro** del contrato **PERROS:**

type Perros struct {

ObjectType string `json:"docType"`

IDPerro int `json:"IDPerro"`

Nombre string `json:"Nombre"`

IDAfijo int `json:"IDAfijo"`

IDSexo int `json:"IDSexo"`

IDPerroMadre int `json:"IDPerroMadre"`

IDPerroPadre int `json:"IDPerroPadre"`

IDRaza int `json:"IDRaza"`

FechaNacimiento string `json:"FechaNacimiento"`

FechaDefuncion string `json:"FechaDefuncion"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

type PerrosPropietarios struct {

ObjectType string `json:"docType"`

IDPerroPropietario int `json:"IDPerroPropietario"`

IDPerro int `json:"IDPerro"`

IDPersona int `json:"IDPersona"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}



En caso contrario devolverá los registros de la solicitud insertados:

type Solicitudes struct {

    ObjectType           string `json:"docType"`

    IDSolicitud          int    `json:"IDSolicitud"`

    TipoSolicitud        string `json:"TipoSolicitud"`

    JSONSolicitud        string `json:"JSONSolicitud"`

    IDPersonaSolicitante int    `json:"IDPersonaSolicitante"`

    EstadoSolicitud      string `json:"EstadoSolicitud"`

    FechaEjecucion       string `json:"FechaEjecucion"`

    FechaAlta            string `json:"FechaAlta"`

    FechaBaja            string `json:"FechaBaja"`

}

type SolicitudesAutorizaciones struct {

    ObjectType      string `json:"docType"`

    IDSolicitud     int    `json:"IDSolicitud"`

    IDPersona       int    `json:"IDPersona"`

    EstadoSolicitud string `json:"EstadoSolicitud"`

    FechaEjecucion  string `json:"FechaEjecucion"`

    FechaAlta       string `json:"FechaAlta"`

    FechaBaja       string `json:"FechaBaja"`

}

donde la autorización del usuario que realiza la solicitud estará registrada y aprobada automáticamente, por lo que no necesitará validar esta acción.

El contrato valida los mismos parámetros que si realizara la acción y no permitirá que se registren una solicitud si existen otra activa, referente a la misma acción.

De esta forma existe la siguiente correlación de funciones y parámetros entre contratos:

PERROS

### **solicitarRegistrarCamada**

* Función **registrarPerro** del contrato **PERROS**

### **solicitarRegistrarPerro**

* Función **registrarPerro** del contrato **PERROS**

### **solicitarRegistrarCambioPropietarioPerro**

* Función **registrarCambioPropietario** del contrato **PERROS**

AFIJOS

### **solicitarRegistrarCambioPropietarioAfijo**

* Función **registrarCambioPropietario** del contrato **AFIJOS**

### **solicitarRegistrarCancelacionAfijo**

* Función **registrarCancelacionAfijo** del contrato **AFIJOS**



El contrato inteligente también incorpora las siguientes funciones:

### **validarSolicitud**

Registrar las validaciones que los usuarios implicados en una solicitud realizan sobre una petición. en el sistema el número o código de microchip insertado subcutáneamente al ejemplar por un veterinario.

En el caso de que todos los usuarios implicados en la solicitud den el visto bueno a solicitud, esta se procederá a ejecutarse automáticamente.

* **Argumentos entrada:**

type tipoValidarSolicitud struct {

IDSolicitud int `json:"IDSolicitud"`

EstadoSolicitud string `json:"EstadoSolicitud"`

}

type TipoSeguridad struct {

    IDPersona int `json:"IDPersona"`

}

|  |  |
| --- | --- |
| El contrato realiza las siguientes validaciones:   * El número de argumentos de entrada sea 2 * El formato JSON de args[0] y args[1] sea valido * El IDSolicitud tenga un valor y corresponda a una solicitud que se encuentre en estado pendiente ( no ejecutada y no caducada) * El EstadoSolicitud tenga un valor valido (APROBADO / RECHAZADO). * La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción |  |

* **Argumentos salida:**

En el caso de que el usuario valide la solicitud y no quede ningún otro usuario por validar el contrato ejecutara automáticamente la solicitud devolviendo los parámetros de la ejecución de la acción solicitada.

En caso contrario devolverá el registros actualizado:

type SolicitudesAutorizaciones struct {

ObjectType string `json:"docType"`

IDSolicitud int `json:"IDSolicitud"`

IDPersona int `json:"IDPersona"`

EstadoSolicitud string `json:"EstadoSolicitud"`

FechaEjecucion string `json:"FechaEjecucion"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

**cargarDatosIniciales**

Introduce en el sistema el historial de las solicitudes que han realizado los diferentes usuarios, definidos en un fichero de texto con el siguiente formato JSON:

type Solicitudes struct {

ObjectType string `json:"docType"`

IDSolicitud int `json:"IDSolicitud"`

TipoSolicitud string `json:"TipoSolicitud"`

JSONSolicitud string `json:"JSONSolicitud"`

IDPersonaSolicitante int `json:"IDPersonaSolicitante"`

EstadoSolicitud string `json:"EstadoSolicitud"`

FechaEjecucion string `json:"FechaEjecucion"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

* **Argumentos entrada:**

NombreFichero string

* **Argumentos salida:**

Devuelve el número de registros insertados:

NumeroRegistros string

Imagen que contiene perro

Descripción generada automáticamente

**cargarDatosIniciales\_Autorizaciones**

Introduce en el sistema el historial de las autorizaciones de las solicitudes que han realizado los diferentes usuarios, definidos en un fichero de texto con el siguiente formato JSON:

type SolicitudesAutorizaciones struct {

ObjectType string `json:"docType"`

IDSolicitud int `json:"IDSolicitud"`

IDPersona int `json:"IDPersona"`

EstadoSolicitud string `json:"EstadoSolicitud"`

FechaEjecucion string `json:"FechaEjecucion"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

* **Argumentos entrada:**

NombreFichero string

* **Argumentos salida:**

Devuelve el número de registros insertados:

NumeroRegistros string

### **Funciones complementarias**

* **querySolicitudes**

Permite a un usuario consultar las solicitudes de un IDPersona y sus estados

### **Funciones comunes**

Funciones de los contratos del sistema incorporadas a este contrato:

* **getQueryResultForQueryString**
* **asignarEstado**
* **borrarEstado**
* **consultarEstado**
* **consultarRangoEstados**
* **ejecutarConsulta**

**Plan de pruebas**

A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

export CHANNEL\_NAME=netcanchannel

export CA\_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-cert.pem

export ORDENER\_URL=orderer.netcan.com:7050

**# SOLICITUDES**

**# solicitarRegistrarCamada / Perro**

peer chaincode invoke -n solicitudes -c '{"function":"solicitarRegistrarCamada","Args":["{\"Perros\":[{\"nombre\":\"PERRO\_uno\",\"IDSexo\":1},{\"nombre\":\"PERRA\_uno\",\"IDSexo\":0}],\"IDPerroMadre\":0,\"IDPerroPadre\":0,\"FechaNacimiento\":\"2020-08-23\",\"Propietarios\":[{\"IDPersona\":666},{\"IDPersona\":999}]}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n solicitudes -c '{"function":"solicitarRegistrarCamada","Args":["{\"Perros\":[{\"nombre\":\"PERRO\_dos\",\"IDSexo\":1},{\"nombre\":\"PERRA\_dos\",\"IDSexo\":1}],\"IDPerroMadre\":1,\"IDPerroPadre\":50,\"FechaNacimiento\":\"2020-08-23\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n solicitudes -c '{"function":"ejecutarConsulta","Args":["{\"selector\": {\"docType\": \"SOLICITUDES\_AUTORIZACIONES\", \"EstadoSolicitud\": \"PENDIENTE\", \"FechaBaja\": { \"$lt\": \"2020-09-15\"}}} ", "{\"IDPersona\":5}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n solicitudes -c '{"function":"validarSolicitud","Args":["{\"IDSolicitud\":2,\"EstadoSolicitud\":\"APROBADO\"}", "{\"IDPersona\":5}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n solicitudes -c '{"function":"solicitarRegistrarCamada","Args":["{\"Perros\":[{\"nombre\":\"PADRE\_tres\",\"IDSexo\":1},{\"nombre\":\"MADRE\_tres\",\"IDSexo\":1}],\"IDPerroMadre\":1,\"IDPerroPadre\":50,\"FechaNacimiento\":\"2021-05-22\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n solicitudes -c '{"function":"validarSolicitud","Args":["{\"IDSolicitud\":3,\"EstadoSolicitud\":\"RECHAZADO\"}", "{\"IDPersona\":5}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME



**# solicitarRegistrarCambioPropietarioPerro**

peer chaincode invoke -n $CC\_NOMBRE\_SOLICITUDES -c '{"function":"solicitarRegistrarCambioPropietarioPerro","Args":["{\"IDPerro\":1,\"Propietarios\":[{\"IDPersona\":1},{\"IDPersona\":2}]}", "{\"IDPersona\":1}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n solicitudes -c '{"function":"validarSolicitud","Args":["{\"IDSolicitud\":4,\"EstadoSolicitud\":\"APROBADO\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# solicitarRegistrarCambioPropietarioAfijo**

peer chaincode invoke -n solicitudes -c '{"function":"solicitarRegistrarCambioPropietarioAfijo","Args":["{\"IDAfijo\":2,\"Propietarios\":[{\"IDPersona\":201},{\"IDPersona\":202},{\"IDPersona\":203},{\"IDPersona\":204}]}", "{\"IDPersona\":5}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n solicitudes -c '{"function":"solicitarRegistrarCambioPropietarioAfijo","Args":["{\"IDAfijo\":2,\"Propietarios\":[{\"IDPersona\":5}]}", "{\"IDPersona\":201}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n solicitudes -c '{"function":"validarSolicitud","Args":["{\"IDSolicitud\":6,\"EstadoSolicitud\":\"APROBADO\"}", "{\"IDPersona\":202}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n solicitudes -c '{"function":"validarSolicitud","Args":["{\"IDSolicitud\":6,\"EstadoSolicitud\":\"APROBADO\"}", "{\"IDPersona\":203}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n solicitudes -c '{"function":"validarSolicitud","Args":["{\"IDSolicitud\":6,\"EstadoSolicitud\":\"APROBADO\"}", "{\"IDPersona\":204}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# solicitarRegistrarCancelacionAfijo"**

peer chaincode invoke -n solicitudes -c '{"function":"solicitarRegistrarCancelacionAfijo","Args":["{\"IDAfijo\":50}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n solicitudes -c '{"function":"solicitarRegistrarCambioPropietarioAfijo","Args":["{\"IDAfijo\":3,\"Propietarios\":[{\"IDPersona\":301},{\"IDPersona\":302}]}", "{\"IDPersona\":123}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# ERRORES SOLICITUDES**

peer chaincode invoke -n solicitudes -c '{"function":"validarSolicitud","Args":["{\"IDSolicitud\":2,\"EstadoSolicitud\":\"APROBADO\"}", "{\"IDPersona\":555}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n solicitudes -c '{"function":"validarSolicitud","Args":["{\"IDSolicitud\":2,\"EstadoSolicitud\":\"APROBADO\"}", "{\"IDPersona\":5}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n $CC\_NOMBRE\_SOLICITUDES -c '{"function":"solicitarRegistrarCambioPropietarioPerro","Args":["{\"IDPerro\":123456,\"Propietarios\":[{\"IDPersona\":1},{\"IDPersona\":2}]}", "{\"IDPersona\":1}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n $CC\_NOMBRE\_SOLICITUDES -c '{"function":"solicitarRegistrarCambioPropietarioPerro","Args":["{\"IDPerro\":123456,\"Propietarios\":[{\"IDPersona\":1765},{\"IDPersona\":23456}]}", "{\"IDPersona\":1234}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n solicitudes -c '{"function":"validarSolicitud","Args":["{\"IDSolicitud\":7,\"EstadoSolicitud\":\"APROBADO\"}", "{\"IDPersona\":259}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# CARGAS INICIALES**

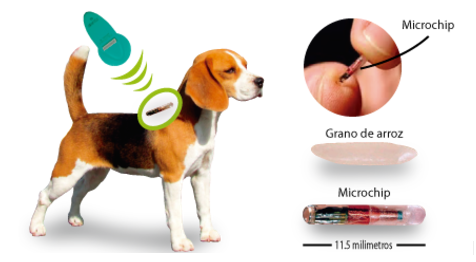
peer chaincode invoke -n solicitudes-c '{"function":"cargarDatosIniciales\_Propietarios","Args":["./json/solicitudes.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n solicitudes-c '{"function":"cargarDatosIniciales\_Propietarios","Args":["./json/autorizaciones.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME



## **MICROCHIP**

Registro de identificación de microchip subcutáneo insertado por los veterinarios que contiene un transpondedor con un código único que permite identificar de manera univoca al ejemplar.



Funciones que incorpora el contrato inteligente:

**registrarMicrochips**

Introduce en el sistema el número o código de microchip insertado subcutáneamente al ejemplar por un veterinario

* **Argumentos entrada:**

type MicrochipsPerros struct {

IDPerro int `json:"IDPerro"`

IDPersonaVeterinario int `json:"IDPersonaVeterinario"`

CODMicrochip string `json:"CODMicrochip"`

}

type TipoSeguridad struct {

    IDPersona int `json:"IDPersona"`

}

|  |  |
| --- | --- |
| El contrato realiza las siguientes validaciones:   * El número de argumentos de entrada sea 2 * El formato JSON de args[0] y args[1] sea valido * Los argumentos Nombre y Propietarios tenga un valor. * El IDPerro tenga un valor y corresponda a un ejemplar vivo. * El IDPersonaVeterinario tenga un valor y corresponda a una persona con un perfil de Veterinario activo. * El CODMicrochip tenga un valor * La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción |  |

* **Argumentos salida:**

Devuelve los registros insertados:

type MicrochipsPerros struct {

ObjectType string `json:"docType"`

IDMicrochipPerro int `json:"IDMicrochipPerro"`

IDPerro int `json:"IDPerro"`

IDPersonaVeterinario int `json:"IDPersonaVeterinario"`

CODMicrochip string `json:"CODMicrochip"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

### **consultarMicrochip**

Consultar los datos del microchip y del ejemplar. La información presentada variara dependiendo de la persona que realiza la consulta.

### **cargarDatosIniciales**

Introduce en el sistema el historial de inserciones de microchips que han tenido los diferentes ejemplares, definidos en un fichero de texto con el siguiente formato JSON:

type MicrochipsPerros struct {

ObjectType string `json:"docType"`

IDMicrochipPerro int `json:"IDMicrochipPerro"`

IDPerro int `json:"IDPerro"`

IDPersonaVeterinario int `json:"IDPersonaVeterinario"`

CODMicrochip string `json:"CODMicrochip"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

* **Argumentos entrada:**

NombreFichero string

* **Argumentos salida:**

Devuelve el número de registros insertados:

NumeroRegistros string

**Imagen que contiene perro, animal, mamífero, café

Descripción generada automáticamenteImagen que contiene perro, animal, mamífero, sostener

Descripción generada automáticamente**

### **Funciones comunes**

Funciones de los contratos del sistema incorporadas a este contrato:

* **getQueryResultForQueryString**
* **asignarEstado**
* **borrarEstado**
* **consultarEstado**
* **consultarRangoEstados**
* **ejecutarConsulta**

**Plan de pruebas**

A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

export CHANNEL\_NAME=netcanchannel

export CA\_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-cert.pem

export ORDENER\_URL=orderer.netcan.com:7050

**# MICROCHIPS**

peer chaincode invoke -n $CC\_NOMBRE\_MICROCHIPS -c '{"function":"registrarMicrochipPerro","Args":["{\"IDPerro\":200,\"IDPersonaVeterinario\":100,\"CODMicrochip\":\"123456789012345\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# ERRORES MICROCHIPS**

peer chaincode invoke -n $CC\_NOMBRE\_MICROCHIPS -c '{"function":"registrarMicrochipPerro","Args":["{\"IDPerro\":200,\"IDPersonaVeterinario\":100000000000,\"CODMicrochip\":\"123456789012345\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n $CC\_NOMBRE\_MICROCHIPS -c '{"function":"registrarMicrochipPerro","Args":["{\"IDPerro\":200,\"IDPersonaVeterinario\":100,\"CODMicrochip\":\"123456789012345\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# CARGAS INICIALES**

peer chaincode invoke -n microchips -c '{"function":"cargarDatosIniciales","Args":["./json/microchips\_perros\_001.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n microchips -c '{"function":"cargarDatosIniciales","Args":["./json/microchips\_perros\_002.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n microchips -c '{"function":"cargarDatosIniciales","Args":["./json/microchips\_perros\_003.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n microchips -c '{"function":"cargarDatosIniciales","Args":["./json/microchips\_perros\_004.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n microchips -c '{"function":"cargarDatosIniciales","Args":["./json/microchips\_perros\_005.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n microchips -c '{"function":"asignarEstado","Args":["MICROCHIPS\_PERROS", "{\"docType\":\"CONTADOR\",\"IDMaximo\":5759}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

## **VACUNAS**

Registro de vacunas administradas por veterinarios.

Los nombres de las vacunas suelen variar según el fabricante y aunque los nombres puedan ser similares (Pentavalente, Hexavalente u Octovalente) su contenido puede variar su contenido con vacunas contra diversas enfermedades tales como, por ejemplo:

* el moquillo canino,
* hepatitis infecciosa
* la leptospirosis
* el parvovirus
* el coronaviru
* la rabia
* la parainfluenza
* la Bordetella bronchisepticala
* la borreliosis o enfermedad de Lyme
* la herpesvirus canino
* la leishmaniasis



Funciones que incorpora el contrato inteligente:

### **registrarVacuna**

Registrar en el sistema el tipo de vacuna administrada al ejemplar, su identificador medicinal, el identificador del colegiado del veterinario, la fecha de administración y la duración de la dosis.

Introduce en el sistema el número o código de microchip insertado subcutáneamente al ejemplar por un veterinario

* **Argumentos entrada:**

type tipoRegistrarVacunaPerroPropietario struct {

       IDVacunaProteccion int    `json:"IDProteccion"`

        FechaBaja      string `json:"FechaBaja"`

    }

    type tipoRegistrarVacunaPerro struct {

        IDPerro             int                      `json:"IDPerro"`

        IDPersonaVeterinario int                     `json:"IDPersonaVeterinario"`

        CODVacuna       string                  `json:"CODVacuna"`

        FechaAlta       string                   `json:"FechaAlta"`

        FechaBaja       string                   `json:"FechaBaja"`

        Protecciones    []tipoRegistrarVacunaPerroPropietario `json:"Protecciones"`

    }

type TipoSeguridad struct {

    IDPersona int `json:"IDPersona"`

}

|  |  |
| --- | --- |
| El contrato realiza las siguientes validaciones:   * El número de argumentos de entrada sea 2 * El formato JSON de args[0] y args[1] sea valido * Los argumentos CODVacuna, FechaAlta y Protecciones tenga un valor. * El IDPerro tenga un valor y corresponda a un ejemplar vivo. * El IDPersonaVeterinario tenga un valor y corresponda a una persona con un perfil de Veterinario activo. * La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción |  |

* **Argumentos salida:**

Devuelve los registros insertados:

type MicrochipsPerros struct {

ObjectType string `json:"docType"`

IDMicrochipPerro int `json:"IDMicrochipPerro"`

IDPerro int `json:"IDPerro"`

IDPersonaVeterinario int `json:"IDPersonaVeterinario"`

CODMicrochip string `json:"CODMicrochip"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

### **obtenerCertificadoVacunaciones (pasaporte)**

Obtener un fichero digital que contiene los datos del ejemplar y su cartilla de vacunación.

### **consultarVacunaciones**

Consultar la cartilla de vacunación del ejemplar. La información presentada variara dependiendo de la persona que realiza la consulta.



### **cargarDatosIniciales**

Introduce en el sistema el historial de vacunaciones que han tenido los diferentes ejemplares, definidos en un fichero de texto con el siguiente formato JSON:

type VacunasPerros struct {

ObjectType string `json:"docType"`

IDVacunaPerro int `json:"IDVacunaPerro"`

IDPerro int `json:"IDPerro"`

IDPersonaVeterinario int `json:"IDPersonaVeterinario"`

CODVacuna string `json:"CODVacuna"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

* **Argumentos entrada:**

NombreFichero string

* **Argumentos salida:**

Devuelve el número de registros insertados:

NumeroRegistros string

### **cargarDatosIniciales\_VacunasPerrosProteccion**

Introduce en el sistema el historial de vacunaciones que han tenido los diferentes ejemplares, definidos en un fichero de texto con el siguiente formato JSON:

type VacunasPerrosProteccion struct {

ObjectType string `json:"docType"`

IDVacunaPerroProteccion int `json:"IDVacunaPerroProteccion"`

IDVacunaPerro int `json:"IDVacunaPerro"`

IDVacunaProteccion int `json:"IDVacunaProteccion"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

* **Argumentos entrada:**

NombreFichero string

* **Argumentos salida:**

Devuelve el número de registros insertados:

NumeroRegistros string

### **cargarDatosIniciales\_VacunasProteccion**

Introduce en el sistema el repositorio de tipos de vacunas más comunes, que determinan el tipo de protección que contiene cada vacuna, que han tenido los diferentes ejemplares, definidos en un fichero de texto con el siguiente formato JSON:

type VacunasProteccion struct {

ObjectType string `json:"docType"`

IDVacunaProteccion int `json:"IDVacunaProteccion"`

Nombre string `json:"Nombre"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

* **Argumentos entrada:**

NombreFichero string

* **Argumentos salida:**

Devuelve el número de registros insertados:

NumeroRegistros string



### **Funciones comunes**

Funciones de los contratos del sistema incorporadas a este contrato:

* **getQueryResultForQueryString**
* **asignarEstado**
* **borrarEstado**
* **consultarEstado**
* **consultarRangoEstados**
* **ejecutarConsulta**

**Plan de pruebas**

A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

export CHANNEL\_NAME=netcanchannel

export CA\_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-cert.pem

export ORDENER\_URL=orderer.netcan.com:7050

**# VACUNAS**

peer chaincode invoke -n vacunas -c '{"function":"registrarVacunaPerro", "Args":[" {\"IDPerro\":6, \"IDPersonaVeterinario\":106, \"CODVacuna\":\"123456789012345\", \"FechaBaja\":\"2022-02-02\", \"Protecciones\":[ {\"IDVacunaProteccion\":1,\"FechaBaja\":\"2022-02-02\"},{\"IDVacunaProteccion\":2,\"FechaBaja\":\"2022-02-02\"} ]}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# ERRORES VACUNAS**

peer chaincode invoke -n vacunas -c '{"function":"registrarVacunaPerro", "Args":[" {\"IDPerro\":6, \"IDPersonaVeterinario\":106, \"CODVacuna\":\"123456789012345\", \"FechaBaja\":\"2022-02-02\", \"Protecciones\":[ {\"IDVacunaProteccion\":1,\"FechaBaja\":\"2022-02-02\"},{\"IDVacunaProteccion\":2,\"FechaBaja\":\"2022-02-02\"} ]}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n vacunas -c '{"function":"registrarVacunaPerro", "Args":[" {\"IDPerro\":6, \"IDPersonaVeterinario\":106, \"CODVacuna\":\"987654321012345\", \"FechaBaja\":\"2022-02-02\", \"Protecciones\":[ {\"IDVacunaProteccion\":1,\"FechaBaja\":\"2022-02-02\"},{\"IDVacunaProteccion\":230,\"FechaBaja\":\"2022-02-02\"} ]}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# CARGAS INICIALES**

peer chaincode invoke -n vacunas -c '{"function":"cargarDatosIniciales\_VacunasProteccion","Args":["./json/vacunas\_proteccion\_001.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

sleep 7s

peer chaincode invoke -n vacunas -c '{"function":"asignarEstado","Args":["VACUNAS\_PERROS\_PROTECCION", "{\"docType\":\"CONTADOR\",\"IDMaximo\":512}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n vacunas -c '{"function":"cargarDatosIniciales","Args":["./json/vacunas\_perros\_001.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

sleep 7s

peer chaincode invoke -n vacunas -c '{"function":"asignarEstado","Args":["VACUNAS\_PROTECCION", "{\"docType\":\"CONTADOR\",\"IDMaximo\":11}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n vacunas -c '{"function":"cargarDatosIniciales\_VacunasPerrosProteccion","Args":["./json/vacunas\_perros\_proteccion\_001.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

sleep 7s

peer chaincode invoke -n vacunas -c '{"function":"asignarEstado","Args":["VACUNAS\_PERROS", "{\"docType\":\"CONTADOR\",\"IDMaximo\":89}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

Imagen que contiene perro, persona, animal, mamífero

Descripción generada automáticamente

## **PERFILES**

Contrato que gestiona el mantenimiento de los diferentes roles especiales de usuario:

* Administradores
* Veterinarios
* Trabajadores de las federaciones caninas

La asignación de los perfiles puede ser consultado desde cualquier contrato, pero solo podrán ser modificados por administradores.



Funciones que incorpora el contrato inteligente:

### **registrarPerfilPersona**

Inserta en el sistema la definición de un nuevo perfil para un usuario.

* **Argumentos entrada:**

type PerfilesPersonasInsertar struct {

    IDPersona       int    `json:"IDPersona"`

    CODPerfil       string `json:"CODPerfil"`

}

type TipoSeguridad struct {

    IDPersona int `json:"IDPersona"`

}

|  |  |
| --- | --- |
| El contrato realiza las siguientes validaciones:   * El número de argumentos de entrada sea 2 * El formato JSON de args[0] y args[1] sea valido * El IDPersona este registrado. * El CODPerfil tenga un valor * La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción (Perfil administrador) |  |

* **Argumentos salida:**

Devuelve los registros insertados:

type PerfilesPersonas struct {

ObjectType string `json:"docType"`

IDPerfilPersona int `json:"IDPerfilPersona"`

IDPersona int `json:"IDPersona"`

CODPerfil string `json:"CODPerfil"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

### **cancelarPerfilPersona**

Cancela en el sistema la definición de un perfil existente para un usuario.

* **Argumentos entrada:**

type PerfilesPersonasCancelar struct {

    IDPersona       int    `json:"IDPersona"`

    CODPerfil       string `json:"CODPerfil"`

}

type TipoSeguridad struct {

    IDPersona int `json:"IDPersona"`

}

|  |  |
| --- | --- |
| El contrato realiza las siguientes validaciones:   * El número de argumentos de entrada sea 2 * El formato JSON de args[0] y args[1] sea valido * El IDPersona este registrado. * El CODPerfil tenga un valor * El IDPersona y CODPerfil tenga definido un perfil activo * La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción (Perfil administrador) |  |

* **Argumentos salida:**

Devuelve los registros insertados:

type PerfilesPersonas struct {

ObjectType string `json:"docType"`

IDPerfilPersona int `json:"IDPerfilPersona"`

IDPersona int `json:"IDPersona"`

CODPerfil string `json:"CODPerfil"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}



### **cargarDatosIniciales**

Introduce en el sistema el historial de perfiles que han tenido los diferentes usuarios, definidos en un fichero de texto con el siguiente formato JSON:

type PerfilesPersonas struct {

ObjectType string `json:"docType"`

IDPerfilPersona int `json:"IDPerfilPersona"`

IDPersona int `json:"IDPersona"`

CODPerfil string `json:"CODPerfil"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

* **Argumentos entrada:**

NombreFichero string

type TipoSeguridad struct {

    IDPersona int `json:"IDPersona"`

}

|  |  |
| --- | --- |
| El contrato realiza las siguientes validaciones:   * El número de argumentos de entrada sea 2 * El formato JSON args[1] sea valido * La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción (Perfil administrador) |  |

* **Argumentos salida:**

Devuelve el número de registros insertados:

NumeroRegistros string

### **Funciones complementarias**

* **asignarEstado\_OnlyAdmin**

Permite a un usuario administrador modificar un estado de la cadena de bloques.

* **borrarEstado\_OnlyAdmin**

Permite a un usuario administrador modificar un estado de la cadena de bloques

* **esAdministrador**

Indica si la persona consultada dispone de un perfil de Administrador activo.

* **esMiembroFederacion**

Indica si la persona consultada dispone de un perfil de miembro en una Federación Canina activo.

* **esVeterinario**

Indica si la persona consultada dispone de un perfil de Veterinario activo.

* **esPerfilPersona**

Indica si la persona consultada dispone de un perfil especifico activo.



### **Funciones comunes**

Funciones de los contratos del sistema incorporadas a este contrato:

* **getQueryResultForQueryString**
* **consultarEstado**
* **consultarRangoEstados**
* **ejecutarConsulta**

**Plan de pruebas**

A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

export CHANNEL\_NAME=netcanchannel

export CA\_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-cert.pem

export ORDENER\_URL=orderer.netcan.com:7050

**# PERFILES**

peer chaincode invoke -n $CC\_NOMBRE\_PERFILES -c '{"function":"registrarPerfilPersona","Args":["{\"IDPersona\":100,\"CODPerfil\":\"ADMINISTRADOR\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n $CC\_NOMBRE\_PERFILES -c '{"function":"cancelarPerfilPersona","Args":["{\"IDPersona**\":100**,\"CODPerfil\":\"ADMINISTRADOR\"}", "{\"IDPersona\":100}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# ERRORES PERFILES**

peer chaincode invoke -n perfiles -c '{"function":"cargarDatosIniciales","Args":["./json/perfiles.json", "{\"IDPersona\":123456}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n $CC\_NOMBRE\_PERFILES -c '{"function":"cancelarPerfilPersona","Args":["{\"IDPersona\":100,\"CODPerfil\":\"ADMINISTRADOR\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n $CC\_NOMBRE\_PERFILES -c '{"function":"cancelarPerfilPersona","Args":["{\"IDPersona\":666,\"CODPerfil\":\"ADMINISTRADOR\"}", "{\"IDPersona\":0}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# CARGAS INICIALES**

peer chaincode invoke -n perfiles -c '{"function":"cargarDatosIniciales","Args":["./json/perfiles.json", "{\"IDPersona\":0}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n perfiles -c '{"function":"asignarEstado\_OnlyAdmin","Args":["PERFILES\_PERSONAS", "{\"docType\":\"CONTADOR\",\"IDMaximo\":48}", "{\"IDPersona\":0}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n perfiles -c '{"function":"cancelarPerfilPersona","Args":["{\"IDPersona\":0,\"CODPerfil\":\"ADMINISTRADOR\"}", "{\"IDPersona\":0}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

## **RAZAS**

Contrato que gestiona el mantenimiento de los diferentes estándares de pureza de raza reconocidos, actualmente cerca de 346 razas.

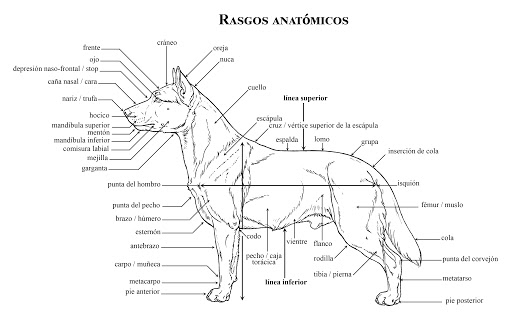
Se clasifican en los siguientes grupos:

* **Grupo 1**: Perros de pastor y perros boyeros (excepto perros boyeros suizos)
* **Grupo 2**: Perros tipo pinscher y schnauzer, molosoides y perros tipo montaña y boyeros suizos
* **Grupo 3**: Terriers
* **Grupo 4**: Teckels
* **Grupo 5**: Tipo spitz y tipo primitivo
* **Grupo 6**: Perros tipo sabueso, perros de rastro (exceptuando lebreles) y razas semejantes.
* **Grupo 7**: Perros de muestra
* **Grupo 8**: Perros cobradores de caza - perros levantadores de caza - perros de agua
* **Grupo 9**: Perros de compañía
* **Grupo 10**: Lebreles



* **Razas Provisionales:** Razas provisionalmente aceptadas
* **Mestizos:** de padres desconocidos, diferente raza o cuyo origen de especie no esté validado.

La definición de las razas y grupos puede ser consultado desde cualquier contrato, pero solo podrán ser modificados por miembros de federaciones caninas.



### **Funciones comunes**

Funciones de los contratos del sistema incorporadas a este contrato:

* **getQueryResultForQueryString**
* **asignarEstado**
* **borrarEstado**
* **consultarEstado**
* **consultarRangoEstados**
* **ejecutarConsulta**

**Plan de pruebas**

A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

export CHANNEL\_NAME=netcanchannel

export CA\_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-cert.pem

export ORDENER\_URL=orderer.netcan.com:7050

**# CARGAS INICIALES**

peer chaincode invoke -n razas -c '{"function":"cargarDatosIniciales\_Grupos","Args":["./json/grupos.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n razas -c '{"function":"asignarEstado","Args":["GRUPOS", "{\"docType\":\"CONTADOR\",\"IDMaximo\":12}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n razas -c '{"function":"cargarDatosIniciales","Args":["./json/razas.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n razas -c '{"function":"asignarEstado","Args":["RAZAS", "{\"docType\":\"CONTADOR\",\"IDMaximo\":346}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME



## **VETERINARIOS**

Contrato que gestiona los datos específicos de un veterinario.

Para la prueba de concepto se ha simplificado los datos a gestionar reduciéndolos al número de colegiado.

En un caso real este contrato podrá contener toda la información que sea necesaria en relación con los veterinarios.

Imagen que contiene perro, persona, interior, coche

Descripción generada automáticamente

Funciones que incorpora el contrato inteligente:

### **registrarColegiaturaPersona**

Inserta en el sistema el registro de una colegiatura de un usuario.

* **Argumentos entrada:**

type ColegiaturasPersonasRegistro struct {

    IDPersona            int    `json:"IDPersona"`

    CODColegiatura       string `json:"CODColegiatura"`

}

type TipoSeguridad struct {

    IDPersona int `json:"IDPersona"`

}

|  |  |
| --- | --- |
| El contrato realiza las siguientes validaciones:   * El número de argumentos de entrada sea 2 * El formato JSON de args[0] y args[1] sea valido * El IDPersona este registrado. * El CODColegiatura tenga un valor y no este registrado ya en el sistema * La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción |  |

* **Argumentos salida:**

Devuelve los registros insertados:

type ColegiaturasPersonas struct {

    ObjectType           string `json:"docType"`

    IDColegiaturaPersona int    `json:"IDColegiaturaPersona"`

    IDPersona            int    `json:"IDPersona"`

    CODColegiatura       string `json:"CODColegiatura"`

    FechaAlta            string `json:"FechaAlta"`

    FechaBaja            string `json:"FechaBaja"`

}

### **cancelarPerfilPersona**

Cancela en el sistema la definición de una colegiatura existente para un usuario.

* **Argumentos entrada:**

type PerfilesPersonasCancelar struct {

    IDPersona       int    `json:"IDPersona"`

    CODPerfil       string `json:"CODPerfil"`

}

type TipoSeguridad struct {

    IDPersona int `json:"IDPersona"`

}

|  |  |
| --- | --- |
| El contrato realiza las siguientes validaciones:   * El número de argumentos de entrada sea 2 * El formato JSON de args[0] y args[1] sea valido * El IDPersona este registrado. * El CODColegiatura tenga un valor * El IDPersona y CODColegiatura tenga definido un perfil activo * La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción |  |

* **Argumentos salida:**

Devuelve los registros insertados:

type ColegiaturasPersonas struct {

    ObjectType           string `json:"docType"`

    IDColegiaturaPersona int    `json:"IDColegiaturaPersona"`

    IDPersona            int    `json:"IDPersona"`

    CODColegiatura       string `json:"CODColegiatura"`

    FechaAlta            string `json:"FechaAlta"`

    FechaBaja            string `json:"FechaBaja"`

}

****

### **cargarDatosIniciales**

Introduce en el sistema el historial de colegiaturas que han tenido los diferentes usuarios, definidos en un fichero de texto con el siguiente formato JSON:

type ColegiaturasPersonas struct {

ObjectType string `json:"docType"`

IDColegiaturaPersona int `json:"IDColegiaturaPersona"`

IDPersona int `json:"IDPersona"`

CODColegiatura string `json:"CODColegiatura"`

FechaAlta string `json:"FechaAlta"`

FechaBaja string `json:"FechaBaja"`

}

* **Argumentos entrada:**

NombreFichero string

* **Argumentos salida:**

Devuelve el número de registros insertados:

NumeroRegistros string

### **Funciones comunes**

Funciones de los contratos del sistema incorporadas a este contrato:

* **getQueryResultForQueryString**
* **asignarEstado**
* **borrarEstado**
* **consultarEstado**
* **consultarRangoEstados**
* **ejecutarConsulta**

**Plan de pruebas**

A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

export CHANNEL\_NAME=netcanchannel

export CA\_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-cert.pem

export ORDENER\_URL=orderer.netcan.com:7050

**# VETERINARIOS**

peer chaincode invoke -n veterinarios -c '{"function":"registrarColegiaturaPersona","Args":["{\"IDPersona\":1,\"CODColegiatura\":\"COVM-1234567890\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n veterinarios -c '{"function":"cancelarColegiaturaPersona","Args":["{\"IDPersona\":1,\"CODColegiatura\":\"COVM-1234567890\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# ERRORES VETERINARIOS**

peer chaincode invoke -n veterinarios -c '{"function":"registrarColegiaturaPersona","Args":["{\"IDPersona\":1,\"CODColegiatura\":\"COVM-1234567890\"}", "{\"IDPersona\":123}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n veterinarios -c '{"function":"registrarColegiaturaPersona","Args":["{\"IDPersCODColegiatura\":\"COVM-1234567890\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n veterinarios -c '{"function":"registrarColegiaturaPersona","Args":["{\"IDPersona\":10000000,\"CODColegiatura\":\"COVM-1234567890\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n veterinarios -c '{"function":"cancelarColegiaturaPersona","Args":["{\"IDPersona\":1,\"CODColegiatura\":\"COVM-NO-EXISTE\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

peer chaincode invoke -n veterinarios -c '{"function":"registrarColegiaturaPersona","Args":["{\"IDPersona\":126,\"CODColegiatura\":\"1234567890\"}", "{\"IDPersona\":6}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

**# CARGAS INICIALES**

peer chaincode invoke -n veterinarios -c '{"function":"cargarDatosIniciales","Args":["./json/veterinarios.json"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

leep 7s

peer chaincode invoke -n veterinarios -c '{"function":"asignarEstado","Args":["VETERINARIOS\_PERSONAS", "{\"docType\":\"CONTADOR\",\"IDMaximo\":21}"]}' -o $ORDENER\_URL --tls --cafile $CA\_FILE -C $CHANNEL\_NAME

# **Otros posibles contratos inteligentes**

A continuación, se especifican algunos contratos más que podrían incorporarse en el fututo a la Blockchain, a modo de ejemplo:

## **TITULOS**

Registra los títulos concedidos por federaciones, asociaciones y clubes por los méritos obtenidos por los ejemplares en explosiones y/o concursos

### solicitarTitulo

Registra la solicitud de un título a una organización en base a los resultados obtenidos en exposiciones y concursos. La validación se realiza de manera automática por el sistema.

### obtenerTitulo

Obtener un fichero digital que certifica el título obtenido por el ejemplar.

### consultarTitulos

Obtiene la relación de títulos que un ejemplar ha obtenido hasta ese momento.



## **EXPOSICIONES Y CONCURSOS**

Registra los resultados de los concursos y exposiciones de Morfología, Trabajo y disciplina, Rastreo y Agility que las diferentes federaciones, asociaciones y club realizan.

### registarShow

Registrar los datos y características de una exposición o concurso *(entidad organizadora, tipo de concurso, ámbito, fecha, lugar, …)*

### registarResultadoShow

Registrar el resultado que un ejemplar ha obtenido al participar en una exposición o concurso, tanto a nivel de calificación como de puesto.

### cagarResultadosShow

Realizar la carga masiva de todos los resultados de una exposición, con el fin de registrar las calificaciones y puestos obtenidos por cada ejemplar.

### consultarResultadosShow

Obtiene información de los resultados y calificaciones de exposiciones y concursos.



El ámbito del proyecto puede expandirse a multitud de gestiones del mundo canino, que hoy día no han podido ser abordadas por el momento de manera global.

Por ejemplo:

## **ADN**

Contrato que gestiona el registro de ADN.

Facilitaría la certificación de autenticad de la linera genealógica y permitiría el estudio genético a partir de los datos obtenidos.

## **ENFERMEDADES / TRATAMIENTOS**

Contrato que gestiona el registro de las enfermedades y tratamientos que los veterinarios realizan sobre los perros, pudiéndose disponerse de un historial clínico para que cualquier veterinario pudiera consultarlo cuando llegue un ejemplar a su consulta.

