private service: IOrganisationService;  
  
 constructor(@inject(TYPES.OrganisationService) service: IOrganisationService){  
 this.service = service  
 }

func CreateOrganisation(stub shim.ChaincodeStubInterface, args []string) peer.Response {  
 utils.Logger.Debug("Create Organisation")  
  
 err := cid.AssertAttributeValue(stub, "app.type", "systemAdmin")  
 if err != nil {  
 errMsg:= "Error creating Organisation. The caller was not a systemAdmin"  
 utils.Logger.Error(errMsg + ": " + err.Error())  
 return shim.Error(errMsg)  
 }  
  
 // check number of arguments  
 if len(args) != 1 {  
 errMsg:= "Create organisation only takes 1 arg"  
 utils.Logger.Error(errMsg)  
 return shim.Error(errMsg)  
 }  
  
 fmt.Printf("Arg 0:\n%s\n", fmt.Sprintf("%s", []byte(args[0])))  
  
 anOrganisation := organisation.Organisation{}  
  
 err = json.Unmarshal([]byte(args[0]), &anOrganisation)  
  
 if err != nil {  
 errMsg:= "Error unmarshaling organisation"  
 utils.Logger.Error(errMsg + ": " + err.Error())  
 return shim.Error(errMsg)  
 }  
 fmt.Printf("Key:\n%s\n", fmt.Sprintf("%s", []byte(anOrganisation.ID)))  
  
 organisationAsBytes, err := stub.GetState(anOrganisation.ID)  
  
 if err != nil {  
 utils.Logger.Error(err)  
 errMsg:= "Failed to get organisation"  
 return shim.Error( errMsg + ": " + err.Error())  
 } else if organisationAsBytes != nil {  
 errMsg:= "This organisation already exists"  
 utils.Logger.Info( errMsg + ": " + anOrganisation.ID)  
 return shim.Error(errMsg + ": " + anOrganisation.ID)  
 }  
  
 err = StoreOrganisation(stub, anOrganisation)  
  
 if err != nil {  
 utils.Logger.Error(err)  
 return shim.Error("Failed to store organisation: " + err.Error())  
 }  
  
 return shim.Success(nil)  
}  
  
func StoreOrganisation(stub shim.ChaincodeStubInterface, organisation organisation.Organisation) error {  
 utils.Logger.Debug("Storing organisation with key: " + organisation.ID)  
  
 organisation.ObjectType = "organisation"  
  
 value2Store, err := json.Marshal(organisation)  
  
 if err != nil {  
 utils.Logger.Error(err)  
 return errors.New("Error when marshaling organisation")  
 } else if value2Store == nil {  
 utils.Logger.Error("Unexpected error organisation is nil")  
 return errors.New("Unexpected error organisation is nil")  
 }  
}