

Chegamos a última parte de nossa aula Blockchain Experience.

Nesta última parte você terá contato com uma versão Early Release do Oracle Autonomous Blockchain Cloud Service, que estará disponível em breve para os nossos clientes.

Você também poderá interagir com o contrato inteligente que possui a implementação de algumas funcionalidades do jogo Monopoly. Para essa interação serão utilizadas as anotações que foram realizadas durante as dinâmicas com o tabuleiro.

Durante o Hands-on você poderá solicitar ajuda ao instrutor e a equipe de apoio para tirar dúvidas e outras orientações.

Para sugestões sobre o conteúdo deste material, entre em contato para fernando.galdino@oracle.com.

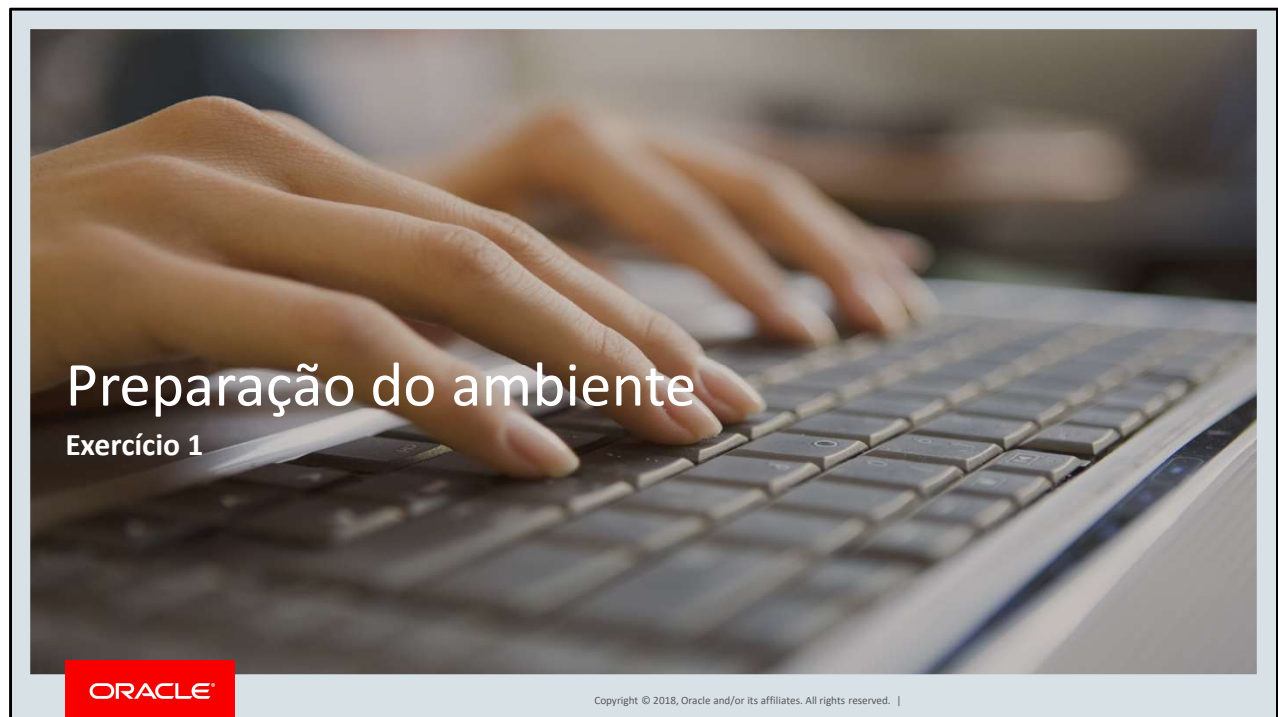
Conteúdo

- Iniciando o OBCS
- Configurando o OBCS
- Conhecendo o smart contract
- Realizando chamadas via API REST
- Realizando chamadas via aplicação

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved. |

Este é o conteúdo que você irá explorar durante a execução das atividades deste exercício.



Dados de acesso Oracle Cloud Trial

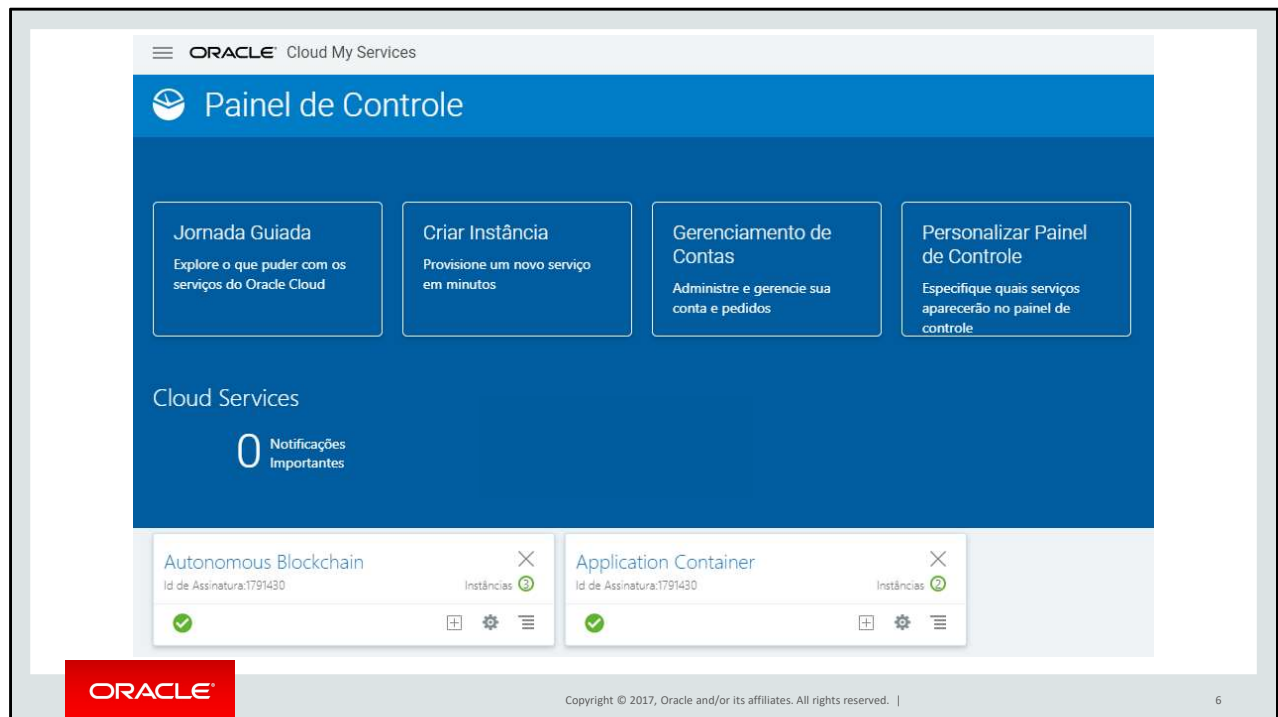
usuário: << E-MAIL ASSOCIADO>>
senha: <<SENHA>>
conta: <<CONTA>>

**SLIDE OCULTO
PARA REFERÊNCIA
APENAS**

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved. |

5

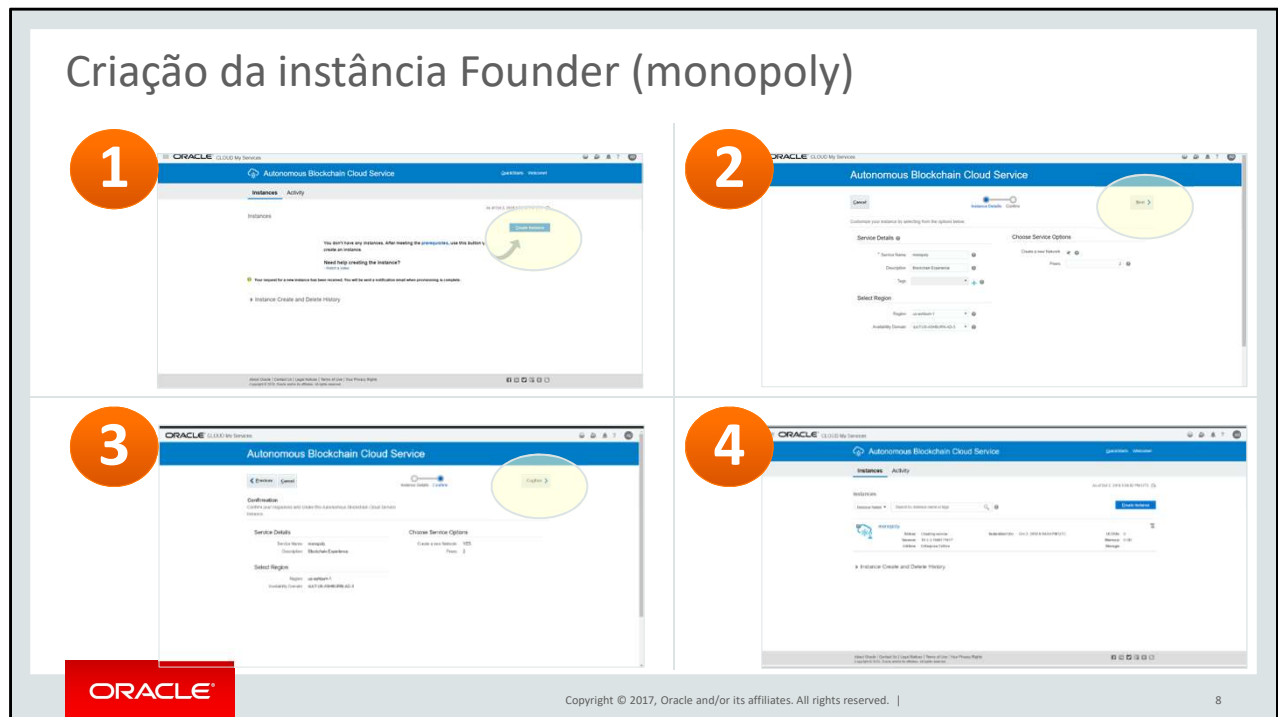


The screenshot displays the Oracle Autonomous Blockchain Cloud Service interface. At the top, there's a blue header with the Oracle logo and the text 'Autonomous Blockchain Cloud Service'. Below this, a navigation bar shows 'Instances' and 'Activity' tabs. A timestamp 'As of Oct 5, 2018 1:46:39 AM UTC' is visible. The main section is titled 'Instances' and includes a search bar with the placeholder 'Search by instance name or tags' and a 'Create Instance' button. Below the search bar, a table lists three instances: 'board2', 'board1', and 'monopoly'. Each instance entry shows its version (18.3.3-1809171617), edition (Enterprise Edition), creation time, and resource allocation (14 OCPUs, 210 GB Memory, and 2,788 GB or 3,838 GB Storage). The Oracle logo is at the bottom left, and a copyright notice is at the bottom center.

Instance Name	Version	Edition	Created On	OCPUs	Memory	Storage
board2	18.3.3-1809171617	Enterprise Edition	Oct 2, 2018 6:59:18 PM UTC	14	210 GB	2,788 GB
board1	18.3.3-1809171617	Enterprise Edition	Oct 2, 2018 6:58:07 PM UTC	14	210 GB	3,888 GB
monopoly	18.3.3-1809171617	Enterprise Edition	Oct 2, 2018 5:54:00 PM UTC	14	210 GB	3,838 GB

Para este exercício devem ser criadas instâncias do Oracle Autonomous Blockchain Cloud Service.

Uma instância deve ser criada para o Founder, que é quem irá criar a rede, a autoridade certificadora e o serviço de ordenação (Ordering Service). Com o Founder criado, deve ser criada uma instância para cada tabuleiro.

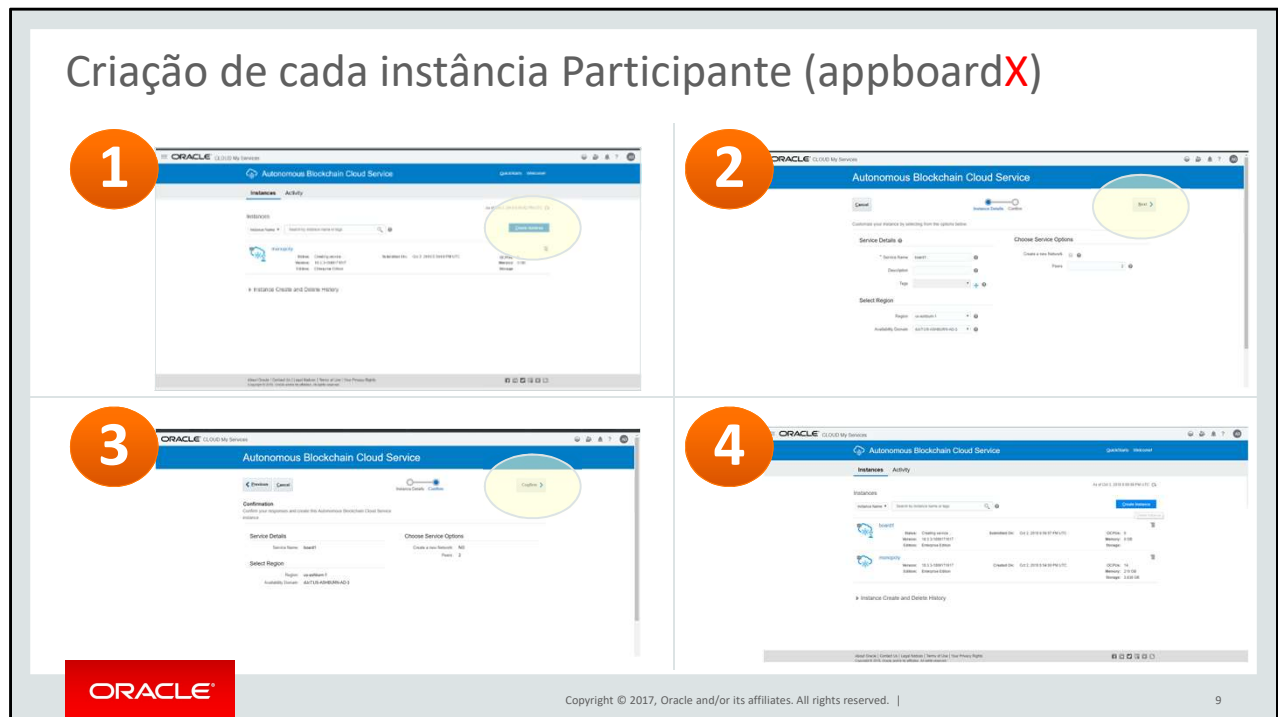


Como criar a rede Founder?

Clicar em Create Instance e fornecer como dados:

- Service Name=monopoly
- Create a new network = checked

Pressionar Next e então, com os dados corretos, pressionar Confirm.



Como criar a instância de cada Participante?

Para cada tabuleiro utilizado, Clicar em Create Instance e fornecer como dados:

- Service Name=board1 (ou outro número, conforme a numeração de cada tabuleiro utilizado)
- Create a new network = not checked

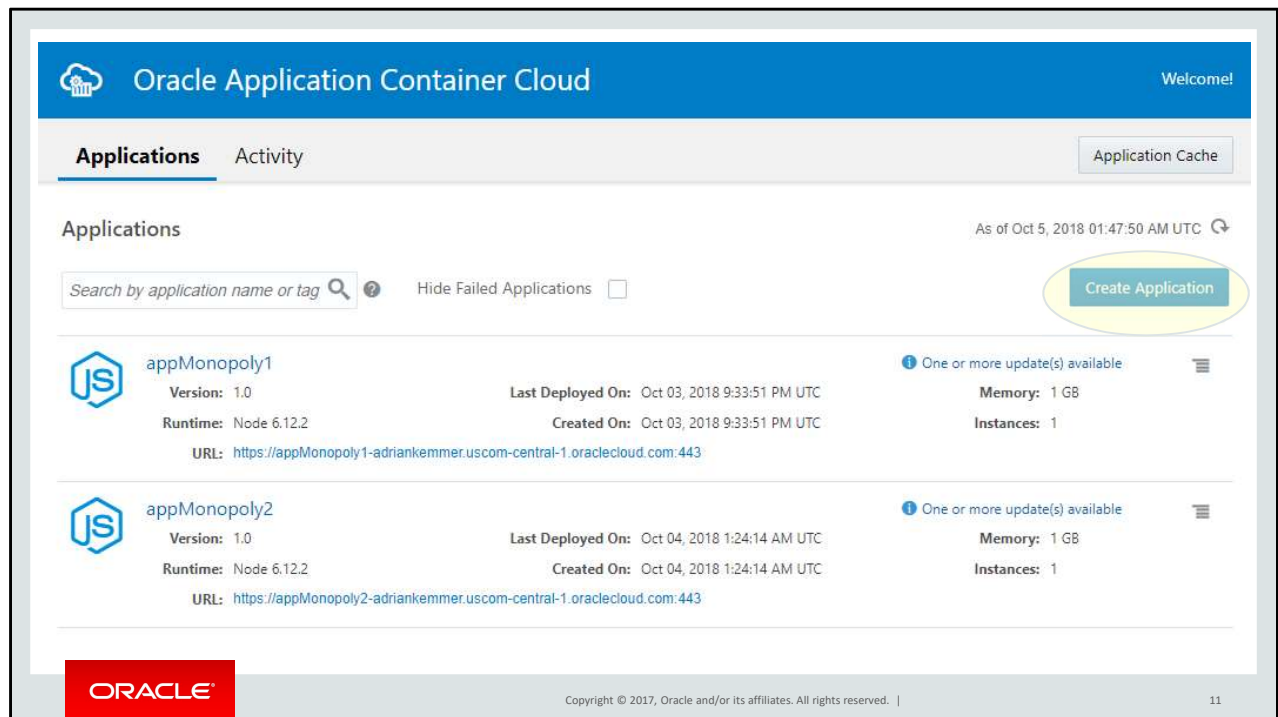
Pressionar Next e então, com os dados corretos, pressionar Confirm.

Configuração do ambiente Founder + Participantes

	Founder	Em cada participante
Exportar certificado da organização		Network / Organization ID / menu Export Certificates
Importar certificados	Acessar Network / Add organizations e fazer o upload dos certificados exportados	
Exportar Ordering Settings		
Importar Ordering Settings		Network / Import Orderer Settings
Exportar nós		Nodes / Export/Import Peers / Export Escolher peer0-1 e peer0-2
Importar nós	Nodes / Export/Import Peers/Import	
Criar new Channel		Create new Channel founderboard1 selecionando os peers peer0-1 e peer0-2.
		Chaincodes / Deploy a new Chaincode Set Anchor Peers
Deploy chaincode		Chaincodes / Deploy Chaincode (Advanced) / Step 1 <ul style="list-style-type: none"> Chaincode Name=monopoly Version=v1 Target peers=peer0-1 e peer0-2 Source=monopoly.zip (adequado ao tabuleiro utilizado)
Instanciar chaincode		Step 2 <ul style="list-style-type: none"> Channel=founderboard1 Peers=peer0-1 e peer0-2
Habilitar REST proxy		Step 3 REST proxy node=restproxy1 Peers=peer0-1 e peer0-2



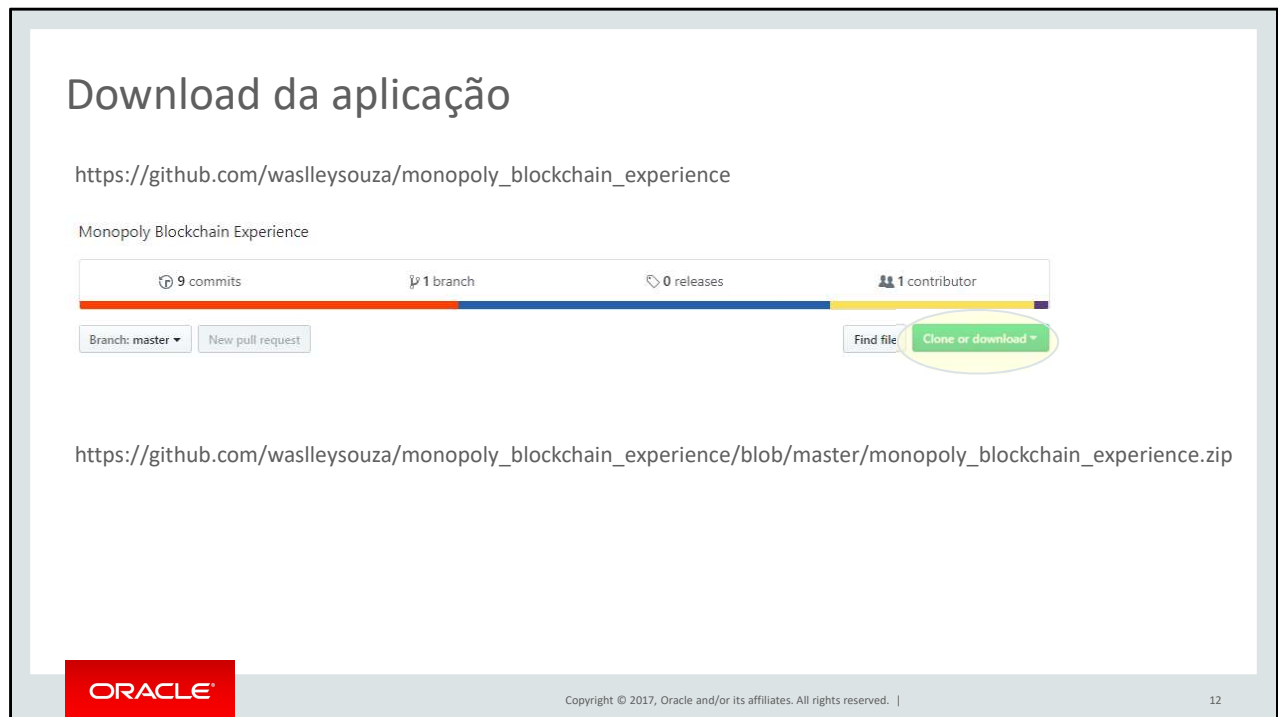
Copyright © 2017, Oracle and/or its affiliates. All rights reserved. |



Será usado o serviço Oracle Application Container Cloud Service para criar a aplicação que foi construída em NodeJS para interagir com as instâncias Blockchain criadas.

Para efeito deste exercício, será atribuído um container NodeJS para cada instância criada no serviço Oracle Autonomous Blockchain. Como sugestão de nomenclatura:

- appboard1 para interação com a instância board1
- appboard2 para interação com a instância board2
- ... e assim por diante.



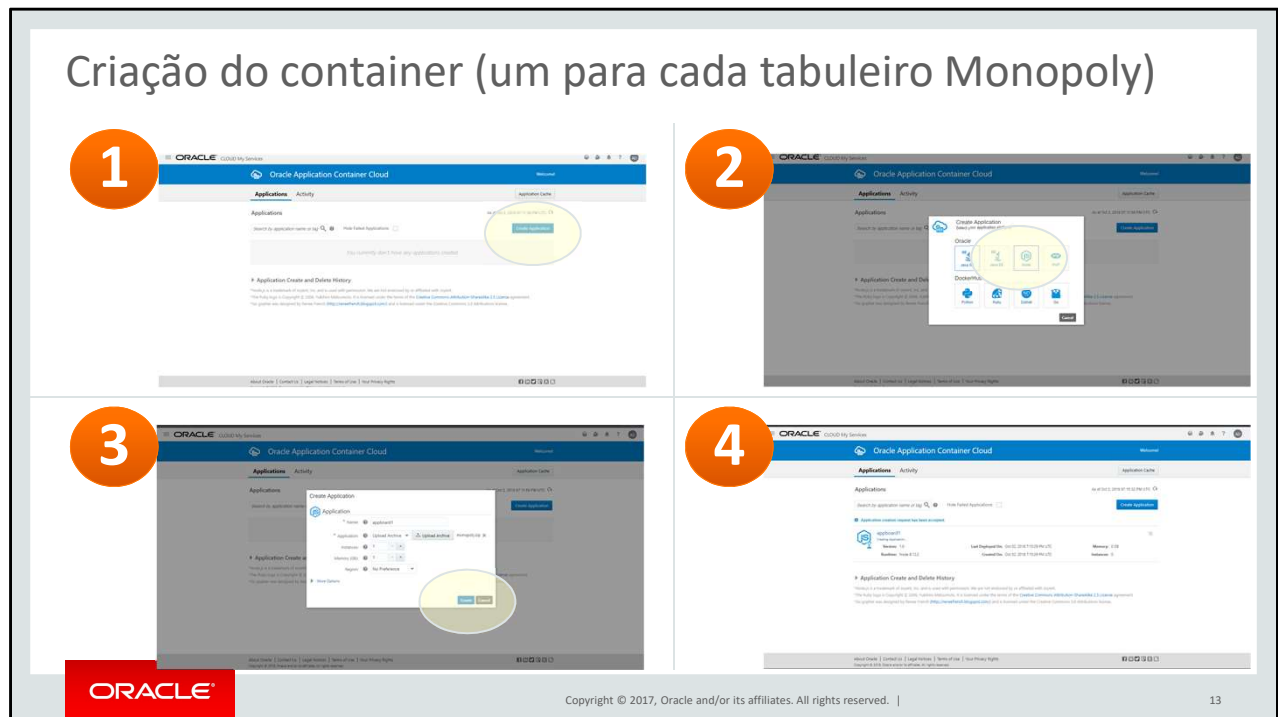
A aplicação para interação com as instâncias Blockchain está armazenada no repositório Github no endereço:

https://github.com/waslleysouza/monopoly_blockchain_experience

Você pode acessar o endereço indicado e clicar em “clone or download” para baixar o conteúdo, ou quaisquer outros comandos git. Depois disso, se pode gerar o zip file da estrutura.

Alternativamente, pode baixar o zip file a ser usado no exercício diretamente a partir do link:

https://github.com/waslleysouza/monopoly_blockchain_experience/blob/master/monopoly_blockchain_experience.zip



Na tela de serviços do Oracle Autonomous Container Cloud Service, clicar em Create Instance.

Na próxima tela, escolher o container NodeJS.

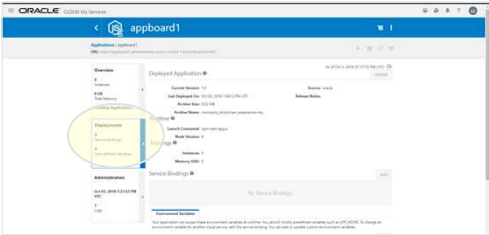
Na sequência definir:

- Name: appboard1 (conforme a sequencia estabelecida)
- Application: selecionar Upload Archive e escolher o arquivo monopoly.zip que foi criado anteriormente
- Instances: 1
- Memory (GB): 1
- Region: No preference

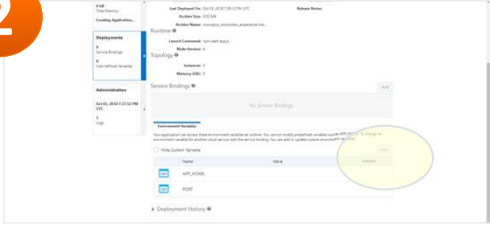
Clicar finalmente em Create e aguardar a criação da instância. Pode-se criar as demais instâncias repetindo-se o processo enquanto as instâncias são inicializadas.

Configuração das variáveis de ambiente

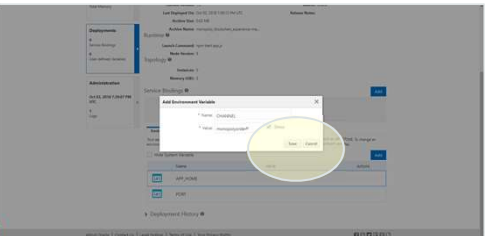
1



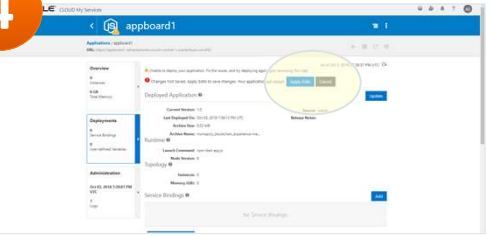
2



3



4













Copyright © 2017, Oracle and/or its affiliates. All rights reserved. |

Para cada aplicação, acessar o painel administrativo.

Procurar pela seção Deployments.

Configuração dos ambientes

appMonopoly1		
	Name	Value
	CHAINCODEEVER	v1
	CHANNEL	founderboard1
	CHAINCODE	monopoly
	INVOCATION_URL	https://board1manager-
	QUERY_URL	https://board1manager-

appMonopoly2		
	Name	Value
	CHANNEL	founderboard2
	CHAINCODE	monopoly
	CHAINCODEEVER	v1
	INVOCATION_URL	https://board2manager-
	QUERY_URL	https://board2manager-



Copyright © 2017, Oracle and/or its affiliates. All rights reserved. |

15

URL de acessos a aplicação

appMonopoly1=https://appmonopoly1-MYCLOUDACCOUNT.uscom-central-1.oraclecloud.com/

appMonopoly2=https://appmonopoly2-MYCLOUDACCOUNT.uscom-central-1.oraclecloud.com/

Os valores de USERNAME e PASSWORD abaixo devem ser atribuídos conforme a conta de cloud atribuída ou conforme usuários associados.

appMonopoly1

CHANNEL=founderboard1

CHAINCODE=monopoly

CHAINCODEEVER=v1

INVOCATION_URL=https://board1manager-

MYCLOUDACCOUNT.blockchain.ocp.oraclecloud.com/restproxy1/bcsgw/rest/v1/transaction/invoke

QUERY_URL=https://board1manager-

MYCLOUDACCOUNT.blockchain.ocp.oraclecloud.com/restproxy1/bcsgw/rest/v1/transaction/query

USERNAME=

PASSWORD=

appMonopoly2

CHANNEL=founderboard2

CHAINCODE=monopoly

CHAINCODEVER=v1

INVOCATION_URL=https://board2manager-

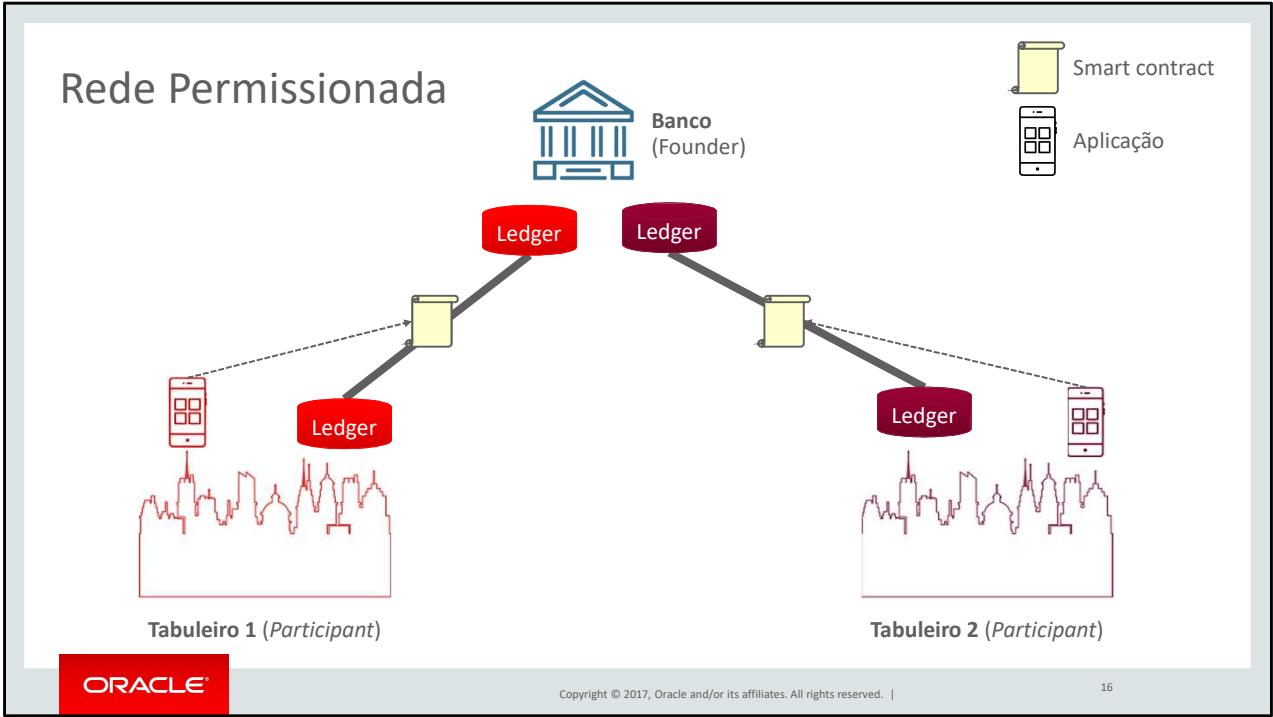
MYCLOUDACCOUNT.blockchain.ocp.oraclecloud.com/restproxy1/bcsgw/rest/v1/trans
action/invocation

QUERY_URL=https://board2manager-

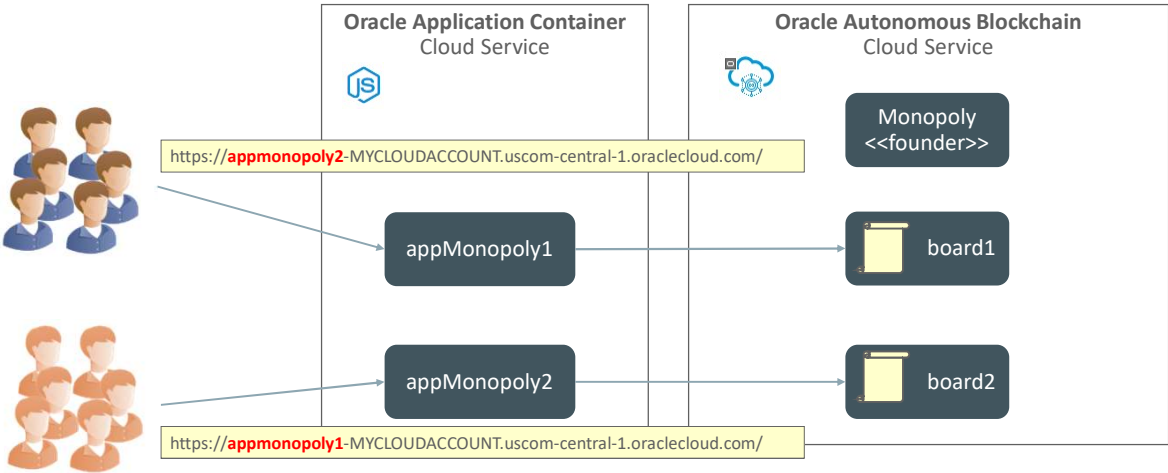
MYCLOUDACCOUNT.blockchain.ocp.oraclecloud.com/restproxy1/bcsgw/rest/v1/trans
action/query

USERNAME=

PASSWORD=



Como acessar?



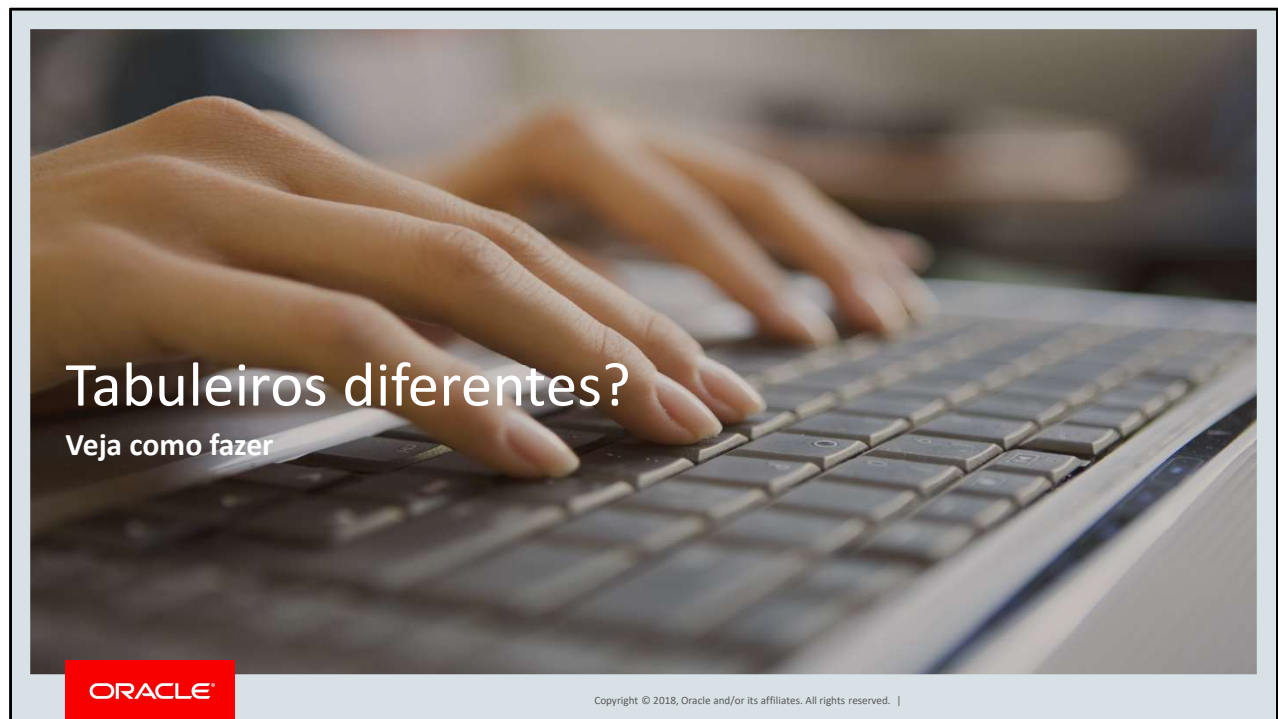
Membros	Propriedades	
<ul style="list-style-type: none"> • Player1 • Player2 • Player3 • Player4 • Player5 • Player6 • Bank 	<ul style="list-style-type: none"> • Ipanema • Leblon • Copacabana • Avenida Brigadeiro Faria Lima • Avenida Presidente Juscelino Kubistcheck • Avenida Engenheiro Luis Carlos Berrini • Avenida Atlântica • Avenida Vieira Souto • Niterói • Avenida Paulista • Rua 25 de Março 	<ul style="list-style-type: none"> • Avenida São João • Praça da Sé • Avenida Sumaré • Avenida Cidade Jardim • Pacaembu • Ibirapuera • Barra da Tijuca • Jardim Botânico • Lagoa Rodrigo de Freitas • Avenida Morumbi • Rua Oscar Freire

ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved. | 18

Para uso das funcionalidades descritas neste documento, você deverá corretamente digitar os nomes dos membros do jogo bem como das propriedades.

Os nomes deverão ser digitados tal como indicados na tabela acima.



Ajustando o Smart Contract

1

Abra o arquivo
monopoly.go

3

Salve com um novo
nome

4

Use este novo contrato
inteligente

2

Atualize a função
getInitialStateProperties

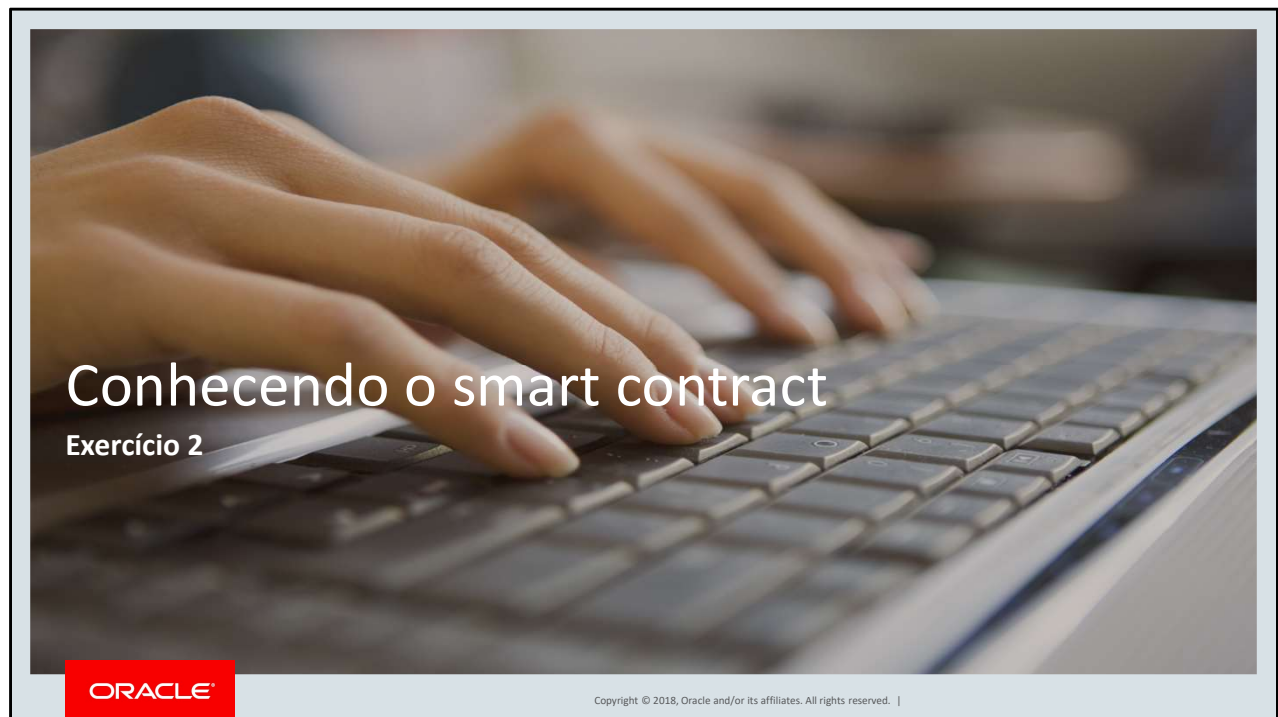
```

336  /* The getInitialStateProperties function
337  .*/
338  */
339  func getInitialStateProperties() []Property {
340      return []Property{
341          Property{Name:"Ipanema", Value:220, Holder:"Bank" },
342          Property{Name:"Leblon", Value:220, Holder:"Bank" },
343          Property{Name:"Copacabana", Value:240, Holder:"Bank" },
344          Property{Name:"Avenida Brigadeiro Faria Lima", Value:200, Holder:"Bank" },
345          Property{Name:"Avenida Presidente Dutra", Value:180, Holder:"Bank" },
346          Property{Name:"Avenida Engenheiro Luís Carlos Berrini", Value:180, Holder:"Bank" },
347          Property{Name:"Avenida Atlântica", Value:160, Holder:"Bank" },
348          Property{Name:"Avenida Vieira Souto", Value:140, Holder:"Bank" },
349          Property{Name:"Hinterland", Value:140, Holder:"Bank" },
350          Property{Name:"Avenida Paulista", Value:120, Holder:"Bank" },
351          Property{Name:"Rua 25 de Março", Value:100, Holder:"Bank" },
352          Property{Name:"Avenida São João", Value:100, Holder:"Bank" },
353          Property{Name:"Praça da Sé", Value:60, Holder:"Bank" },
354          Property{Name:"Avenida Sumaré", Value:60, Holder:"Bank" },
355          Property{Name:"Avenida Cidade Jardim", Value:200, Holder:"Bank" },
356          Property{Name:"Paseo", Value:200, Holder:"Bank" },
357          Property{Name:"Israpuere", Value:200, Holder:"Bank" },
358          Property{Name:"Barra da Tijuca", Value:300, Holder:"Bank" },
359          Property{Name:"Jardim Botânico", Value:300, Holder:"Bank" },
360          Property{Name:"Lagoa Rodrigo de Freitas", Value:320, Holder:"Bank" },
361          Property{Name:"Avenida Norumbé", Value:350, Holder:"Bank" },
362          Property{Name:"Rua Oscar Freire", Value:400, Holder:"Bank" },
363      }
364  }

```

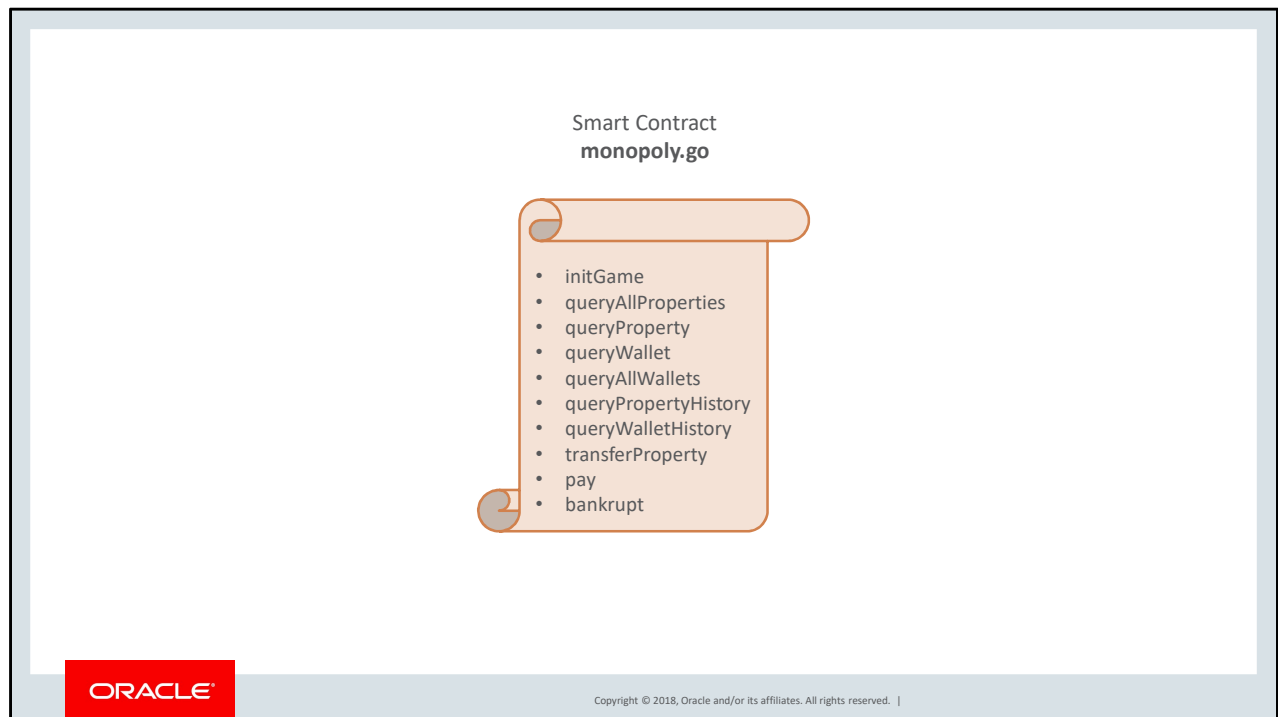
ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved. |



Nesta seção você poderá conhecer um pouco mais sobre como é um Smart Contract na plataforma Hyperledger, que é a base do Oracle Autonomous Blockchain Cloud Service.

O contrato inteligente foi codificado usando a linguagem Go.



O contrato inteligente usado neste exercício pode ser encontrado em:

<https://github.com/maxgaldinus/monopoly/blob/master/chaincode/monopoly.go>

Ele possui seis métodos que realizam a consulta aos dados do ledger:

- queryAllProperties
- queryProperty
- queryWallet
- queryAllWallets
- queryPropertyHistory
- queryWalletHistory

Também possui os métodos que realizam operações que mudam o estado no ledger, como a transferência de propriedades e pagamentos:

- transferProperty
- pay
- bankrupt

Além dos métodos citados acima, ainda contempla o método `initGame`, que é usado para inicializar o contrato inteligente quando necessário.



Esta é a estrutura inicial de um contrato inteligente escrito em Go.

Note algumas bibliotecas sendo importadas, principalmente as que fazem parte do Hyperledger Fabric:

github.com/hyperledger/fabric/core/chaincode/shim
github.com/hyperledger/fabric/protos/peer

Note também a definição de estruturas necessárias para armazenamento das informações. Além da estrutura Property indicada, existe também a estrutura Wallet, que são necessárias para este exercício.

```

65  /**
66  * The Invoke method
67  * called when an application requests to run the Smart Contract
68  * The app also specifies the specific smart contract function to call with args
69  */
70  func (s *SmartContract) Invoke(APIstub shim.ChaincodeStubInterface) sc.Response {
71      // Retrieve the requested Smart Contract function and arguments
72      function, args := APIstub.GetFunctionAndParameters()
73
74      // Route to the appropriate handler function to interact with the ledger appropriately
75      if function == "queryAllProperties" {
76          return s.queryAllProperties(APIstub)
77      } else if function == "queryProperty" {
78          return s.queryProperty(APIstub, args)
79      } else if function == "queryWallet" {
80          return s.queryWallet(APIstub)
81      } else if function == "queryAllWallets" {
82          return s.queryAllWallets(APIstub)
83      } else if function == "queryPropertyHistory" {
84          return s.queryPropertyHistory(APIstub)
85      } else if function == "queryWalletHistory" {
86          return s.queryWalletHistory(APIstub)
87      } else if function == "initGame" {
88          return s.initGame(APIstub)
89      } else if function == "transferProperty" {
90          return s.transferProperty(APIstub, args)
91      } else if function == "pay" {
92          return s.pay(APIstub)
93      } else if function == "bankrupt" {
94          return s.bankrupt(APIstub)
95      }
96
97      return shim.Error("Invalid Smart Contract function name.")
98  }

```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved. |

De acordo com o Hyperledger Fabric, as funcionalidades do contrato inteligente devem ser acessadas a partir da função Invoke. Por essa razão a identificação da função e o roteamento para o código correto são realizados aqui.

```
153 /**
154  * The queryWallet method
155  *
156  */
157 func (s *SmartContract) queryWallet(APIStub shim.ChaincodeStubInterface) sc.Response {
158     var wallet string
159     var err error
160
161     _, args := APIStub.GetFunctionAndParameters()
162
163     if len(args) != 1 {
164         return shim.Error("Incorrect number of arguments. Expecting 1.")
165     }
166
167     wallet = args[0]
168     walletAsBytes, err := APIStub.GetState(wallet)
169     if err != nil {
170         return shim.Error(fmt.Sprintf("Failed to get wallet for %s", wallet))
171     } else if walletAsBytes == nil {
172         return shim.Error(fmt.Sprintf("Wallet does not exist: %s", wallet))
173     }
174     return shim.Success(walletAsBytes)
175 }
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved. |

Este é o código referente ao método queryWallet que tem como objetivo resgatar o valor atual, registrado no ledger, da carteira de um dos jogadores.

```

460  /**
461   * The pay Method
462   * pay(from, to, value)
463   * from is the source wallet
464   * to is the target wallet
465   *
466   */
467 func (s *SmartContract) pay(APIStub shim.ChaincodeStubInterface) sc.Response {
468     _, args := APIStub.GetFunctionAndParameters()
469
470     if len(args) != 3 {
471         return shim.Error("Incorrect number of arguments for pay. Expecting 3 - (from, to, value)")
472     }
473     value, err := strconv.Atoi(args[2])
474     if err != nil {
475         return shim.Error("Invalid value for payment.")
476     }
477     errorMsg := doPayment(APIStub, args[0], args[1], value)
478     if errorMsg != "" {
479         return shim.Error(errorMsg)
480     }
481     return shim.Success(nil);
482 }

```



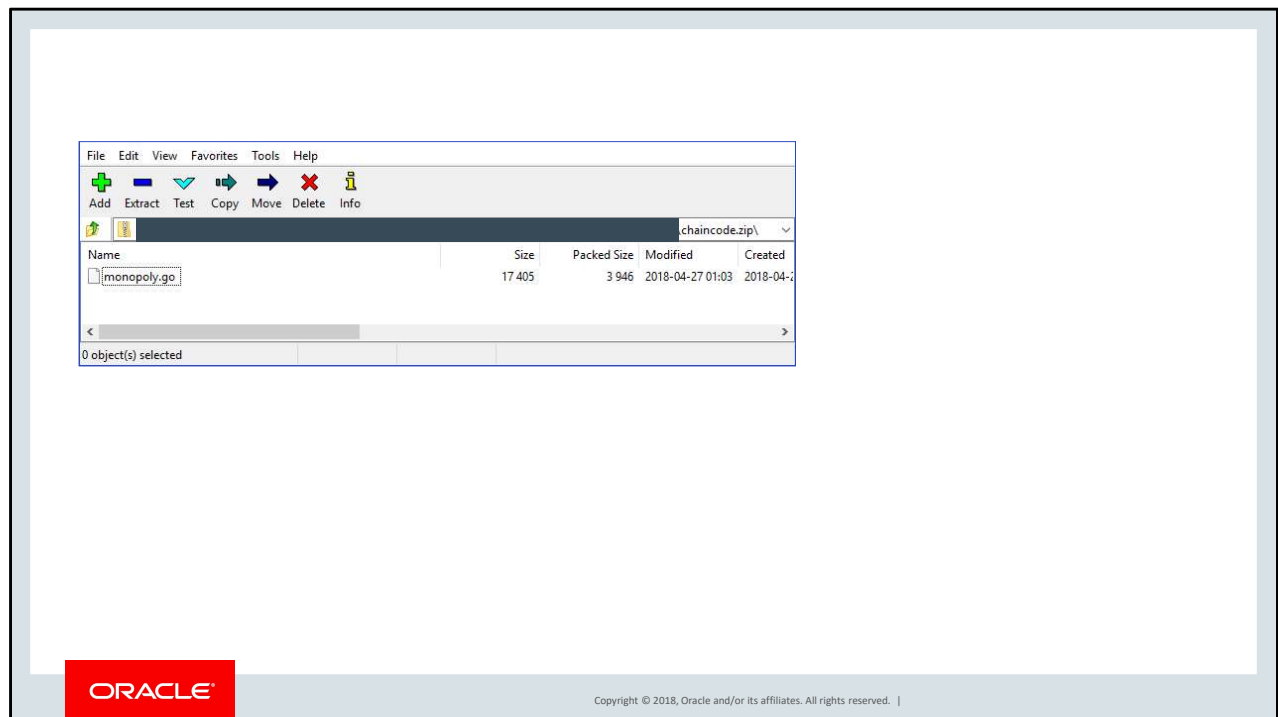
Copyright © 2018, Oracle and/or its affiliates. All rights reserved. |

De forma similar, este é o método pay.

Seu objetivo é realizar a transferência de dinheiro de uma parte para outra.

Para isso são realizadas leituras no ledger para se ter consistência das informações.

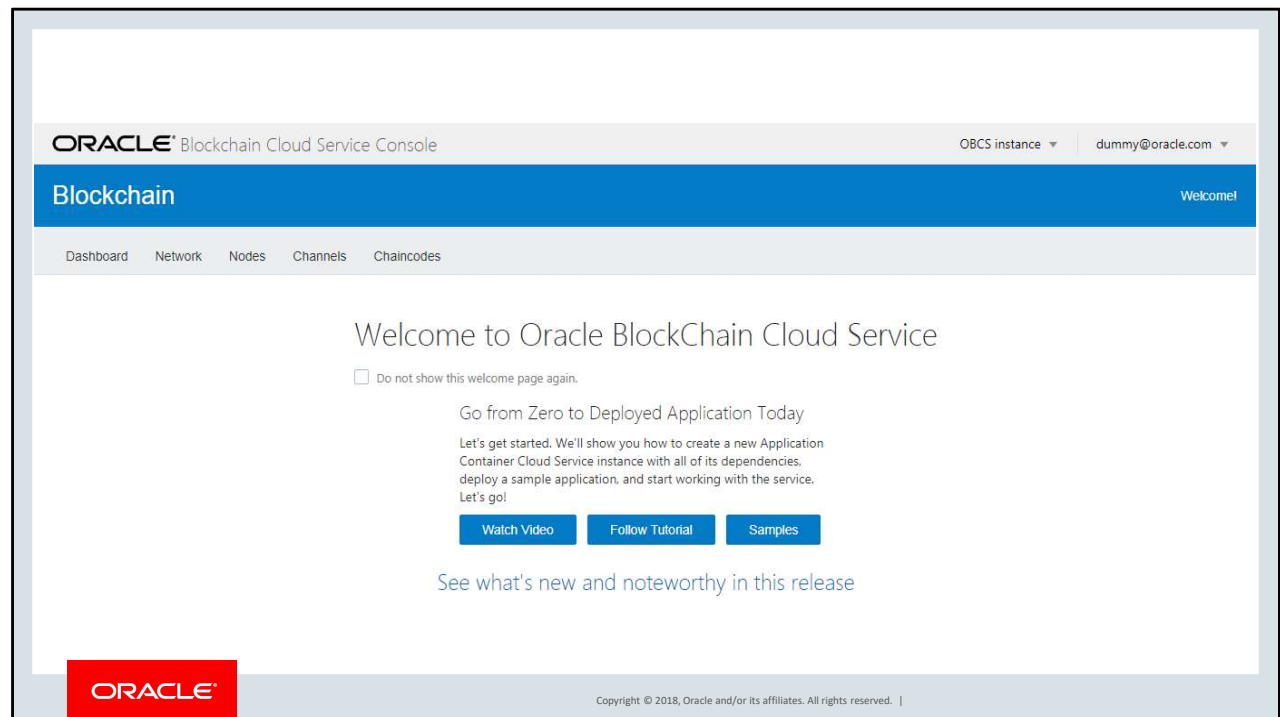
Após isso, o ledger é atualizado de modo a refletir a transferência necessária.



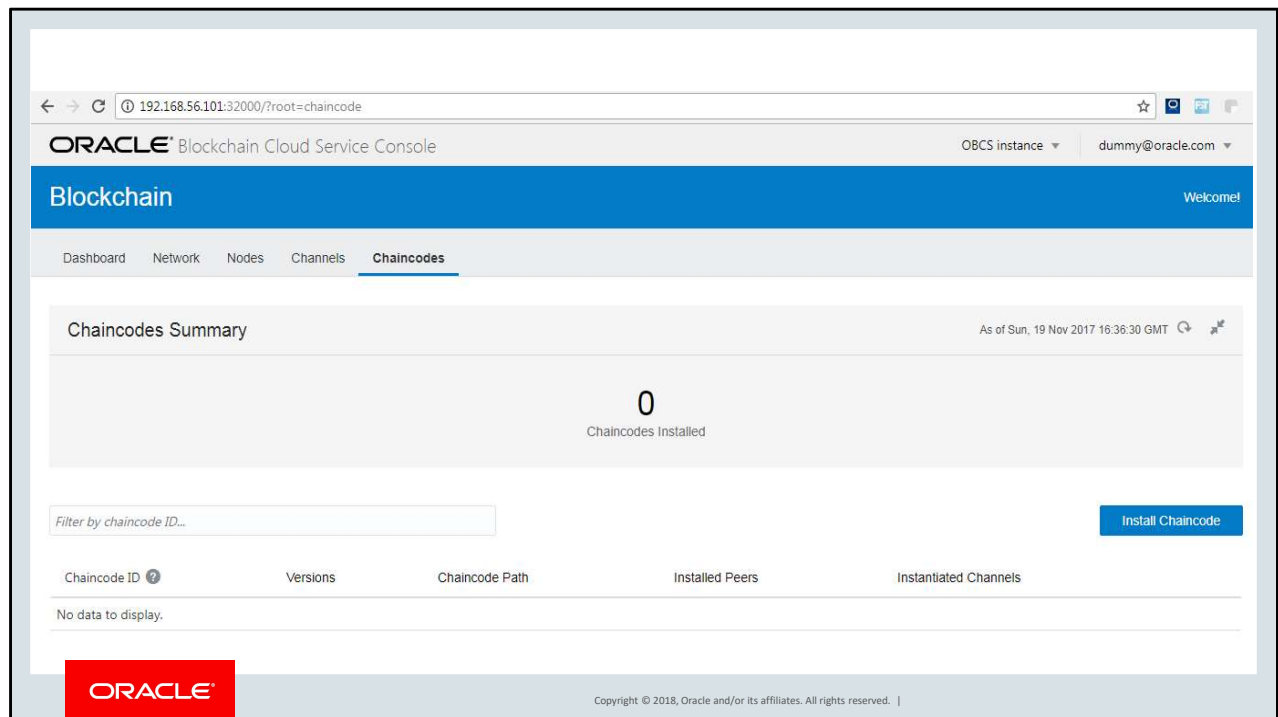
Para implantação (deployment) no Oracle Autonomous Blockchain Cloud Service é necessário que o arquivo monopoly.go esteja dentro de um arquivo .zip.

No caso, chaincode.zip foi o nome escolhido.

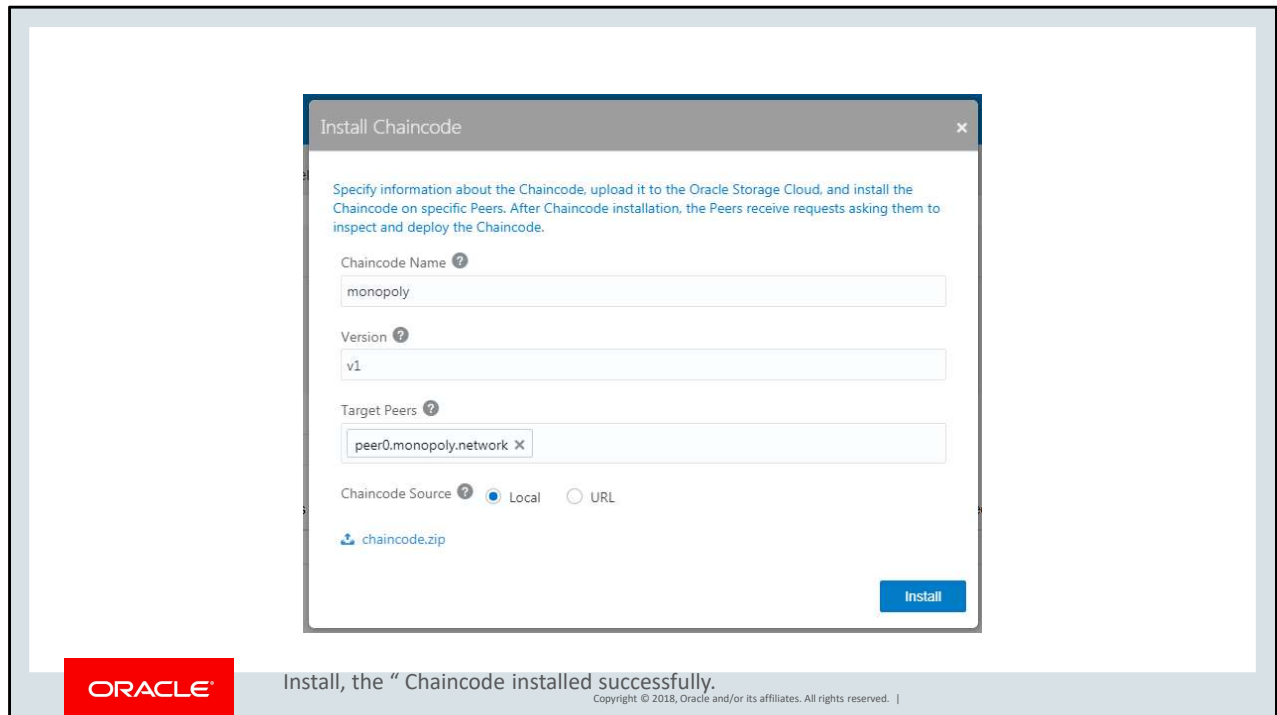




Você terá então acesso a interface abaixo. Neste exercício você irá navegar pelas opções Dashboard, Network, Nodes, Channels e Chaincodes conforme as instruções a seguir.



Para instalar o contrato inteligente acesse a opção Chaincodes e a seguir selecione Install Chaincode.



Install Chaincode

Specify information about the Chaincode, upload it to the Oracle Storage Cloud, and install the Chaincode on specific Peers. After Chaincode installation, the Peers receive requests asking them to inspect and deploy the Chaincode.

Chaincode Name [?]
monopoly

Version [?]
v1

Target Peers [?]
peer0.monopoly.network ✕

Chaincode Source [?] ☒ Local ☐ URL

[chaincode.zip](#)

Install

ORACLE Install, the "Chaincode installed successfully."
Copyright © 2018, Oracle and/or its affiliates. All rights reserved. |

Preencha os dados como indicado:

Chaincode Name = monopoly

Version = v1

Target peers = peer0.monopoly.network

Chaincode Source = Local (zip file chaincode.zip provido para este laboratório)

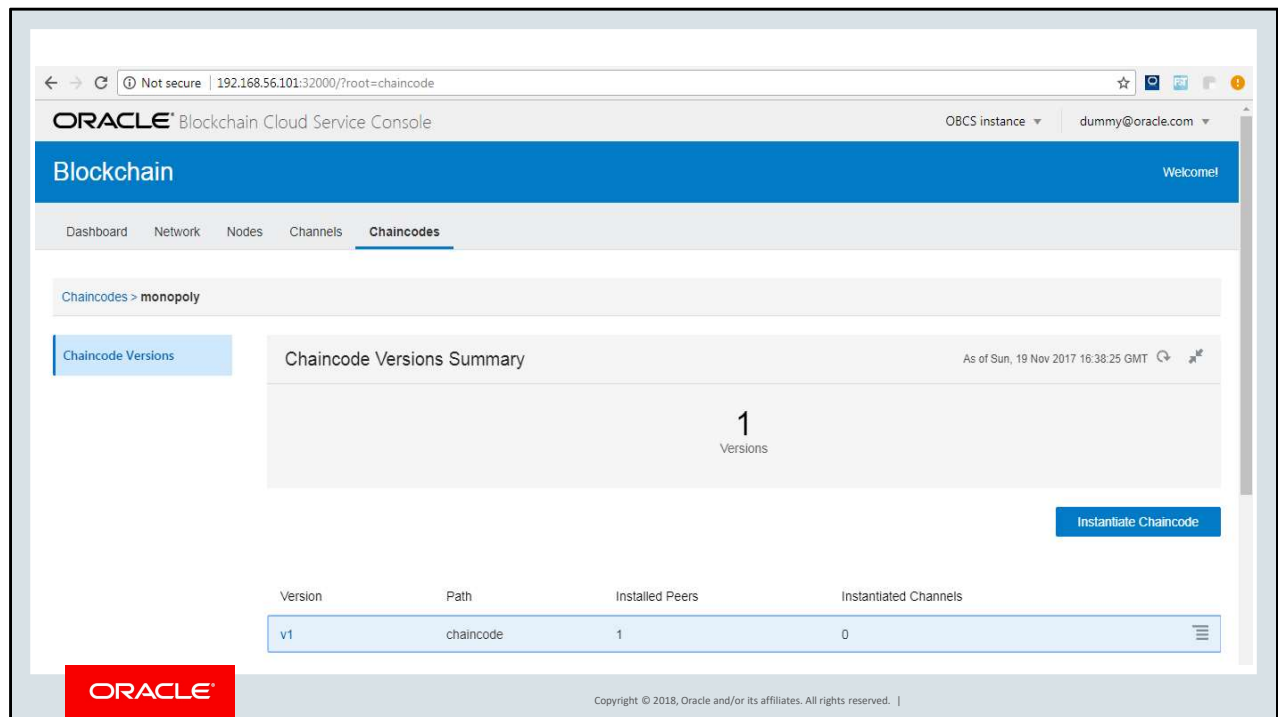
Ao clicar em Install você deverá receber a mensagem **Chaincode installed successfully.**

The screenshot displays the Oracle Blockchain Cloud Service Console interface. At the top, the header includes the Oracle logo, the text 'Blockchain Cloud Service Console', and user information: 'OBSC instance' and 'dummy@oracle.com'. Below the header is a blue navigation bar with the word 'Blockchain' and a 'Welcome!' message. A secondary navigation bar contains links for 'Dashboard', 'Network', 'Nodes', 'Channels', and 'Chaincodes', with 'Chaincodes' being the active tab. The main content area is titled 'Chaincodes Summary' and shows a large number '1' indicating 'Chaincodes Installed'. Below this is a filter input field labeled 'Filter by chaincode ID...' and a blue button labeled 'Install Chaincode'. A table lists the installed chaincodes with columns: 'Chaincode ID', 'Versions', 'Chaincode Path', 'Installed Peers', and 'Instantiated Channels'. The table contains one entry with the ID 'monopoly' and version '1'. The Oracle logo is visible in the bottom left corner, and a copyright notice 'Copyright © 2018, Oracle and/or its affiliates. All rights reserved.' is in the bottom right.

Chaincode ID	Versions	Chaincode Path	Installed Peers	Instantiated Channels
monopoly	1			

O próximo passo é instanciar o contrato inteligente em um Channel. Para isso, ainda na aba Chaincode, clique no chaincode que acabou de ser carregado para o ambiente.

Ao clicar em monopoly, você será direcionado para uma página onde você verá a versão do chaincode, nos Peers onde ele foi instalado e nos Channels.



Como esta é a primeira vez, nenhum Channel está selecionado. Para isso selecione a linha da versão 1.0 (v1) e clique no botão Instantiate Chaincode.

Instantiate Chaincode

Specify the information required to instantiate the Chaincode to the Channel and for the Peers you specify. After you instantiate the Chaincode, the orderer and peer nodes can use it for transactions.

Channel [?]
monopolyorderer

Peers [?]
peer0.monopoly.network

Initial Parameter [?]
["args":["a","100","b","200"]]

▶ Endorsement Policy

▶ Transient Map

Instantiate

Select who you want to use the Chaincode. Choose one peer. The Chaincode must be installed on and endorsed by the peer that you choose.

ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved. |

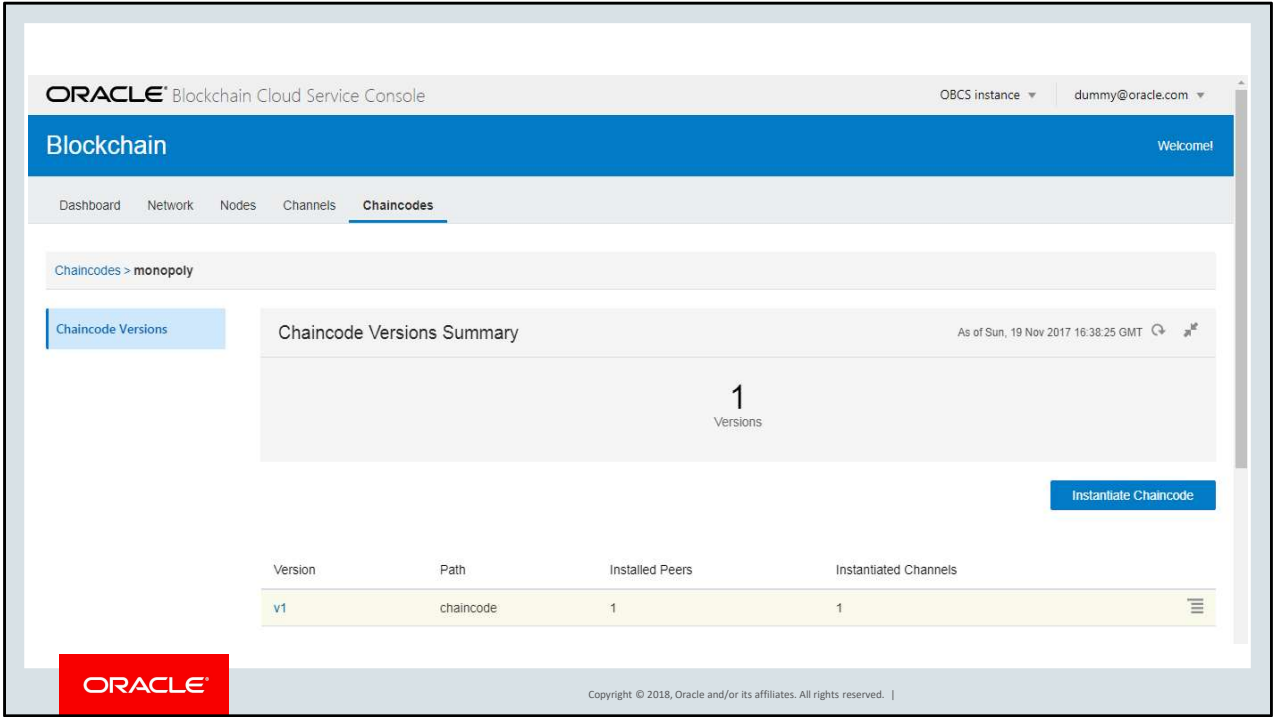
Como indicado, selecione:

Channel = monopolyorderer

Peers = peer0.monopoly.network

Initial parameter = pode deixar os valores apresentados. Eles não são necessários para o chaincode

Ao clicar em Instantiate, o chaincode será instanciado.



Nesta tela podemos verificar que o chaincode foi instanciado.

Blockchain Welcome!

Dashboard Network **Nodes** Channels Chaincodes

Nodes Summary As of Sun, 19 Nov 2017 16:41:17 GMT

5 Nodes | 2 Peers | 1 Orderers | 1 CA | 1 REST Proxy

Filter by Node Name... Status: All Type: All + - ? Add Node

Name	Route	Type	Status
peer0.monopoly.network	grpc://peer0.monopoly.network:7051	Peer	up
peer1.monopoly.network	grpc://peer1.monopoly.network:7051	Peer	up
orderer.monopoly.network	grpc://orderer.monopoly.network:7050	Orderer	up
fabricca0.monopoly.network	fabricca0.monopoly.network:7054	CA	up
gateway0.monopoly.network	https://192.168.56.101:7000	REST Proxy	up

ORACLE Copyright © 2018, Oracle and/or its affiliates. All rights reserved. |

O próximo passo é ativar o acesso via API REST fornecido pelo Oracle Blockchain Cloud Service.

Para isso selecione a aba Nodes. Terão 5 (cinco) nós ativos, conforme ilustra a imagem.

Para cada linha, a direita da coluna Status, existe um menu suspenso que pode ser ativado ao clicar nele. Faça isso para `gateway0.monopoly.network` e selecione a opção Editar.

orderer.monopoly.network	grpc://orderer.monopoly.network:7050	Orderer	up	Edit
fabricca0.monopoly.network	fabricca0.monopoly.network:7054	CA	up	Remove
gateway0.monopoly.network	https://192.168.56.101:7000	REST Proxy	up	

Configure

REST Proxy Name: gateway0.monopoly.network

Proposal Wait Time (s): 60

Transaction Wait Time (s): 30

Select which chaincode on the channels are exposed, as well as which are the endorsing peers for the chaincode:

monopolyorderer monopolyv1

peer0.monopoly.network peer1.monopoly.network

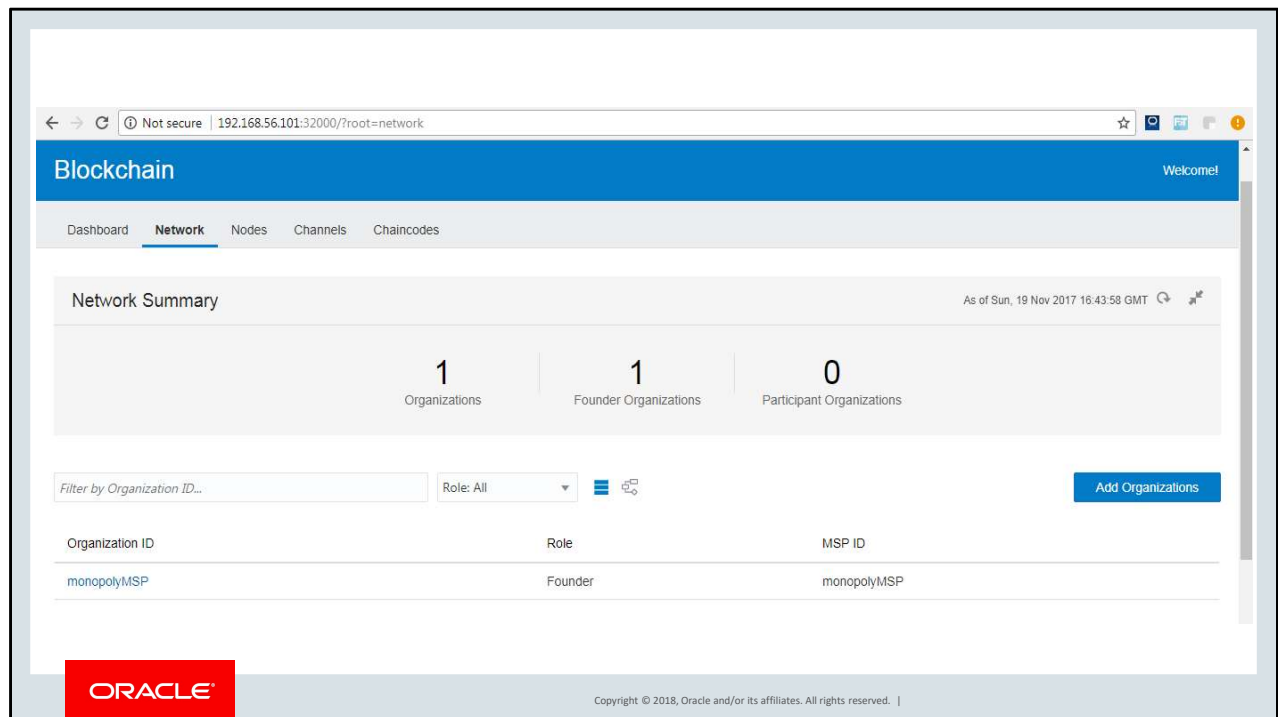
Submit

ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved. |

Na tela de configuração deixe como apresentado e clique em Submit.

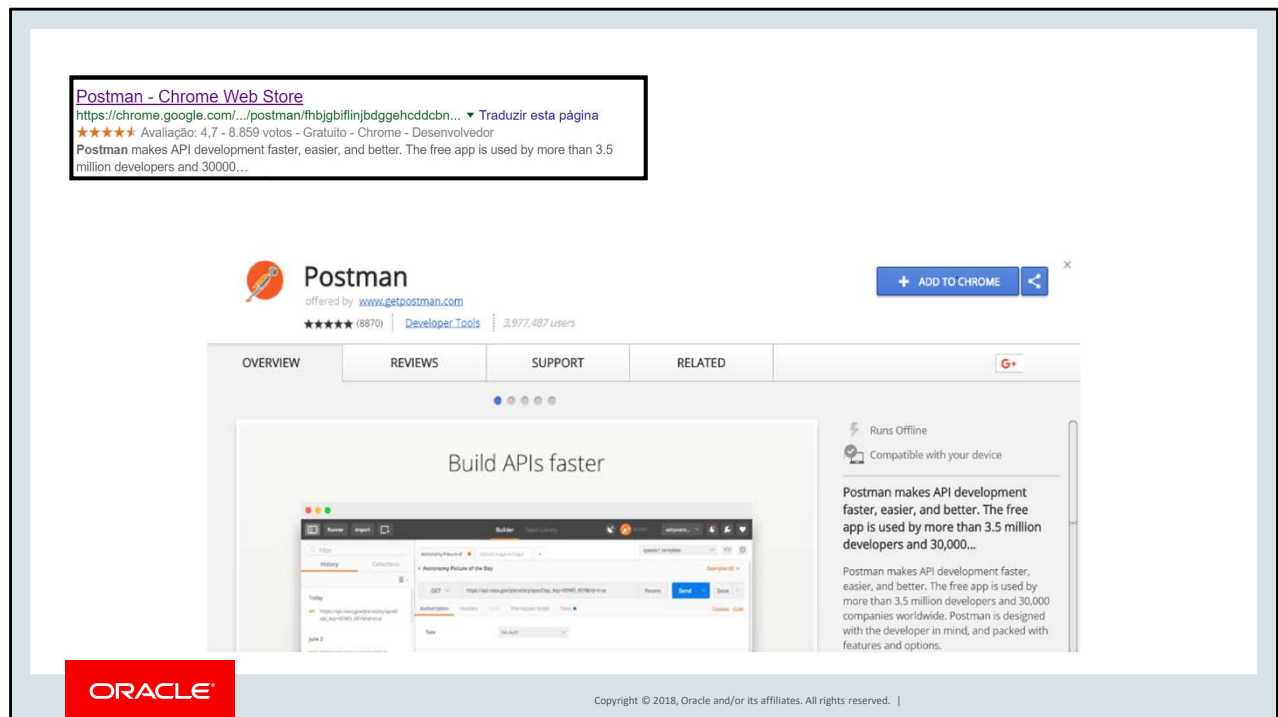
A partir desse ponto você poderá efetuar chamadas REST para o Oracle Blockchain Cloud Service.



Neste exercício teremos apenas o membro Fundador da blockchain.

Num cenário mais elaborado poderíamos ter outros membros participantes: Banco, Governo, Polícia, Casa da Moeda, dentre outros.





Instalando Postman

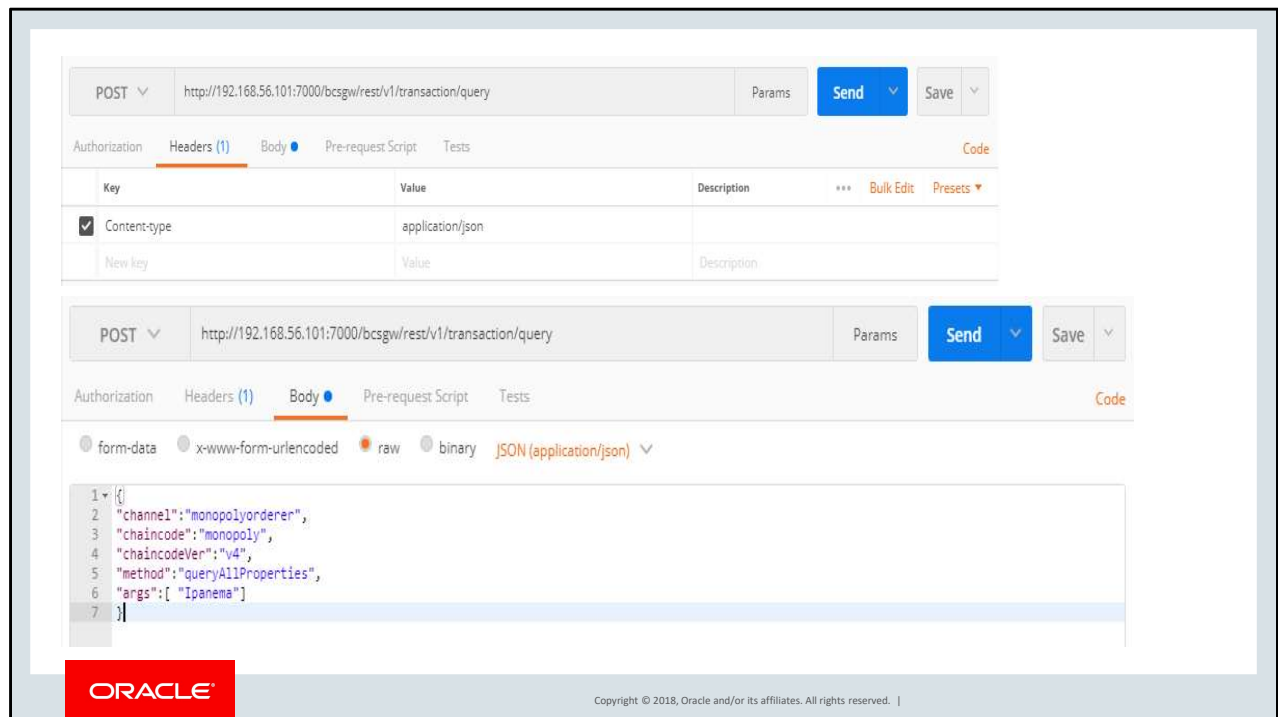
Abra o seu navegador Google Chrome

Busque por “Postman Chrome”

Clique em “Add to Chrome”

Após instalar, acesse “chrome://apps/” em seu Google Chrome e clique em Postman para abrir a aplicação.

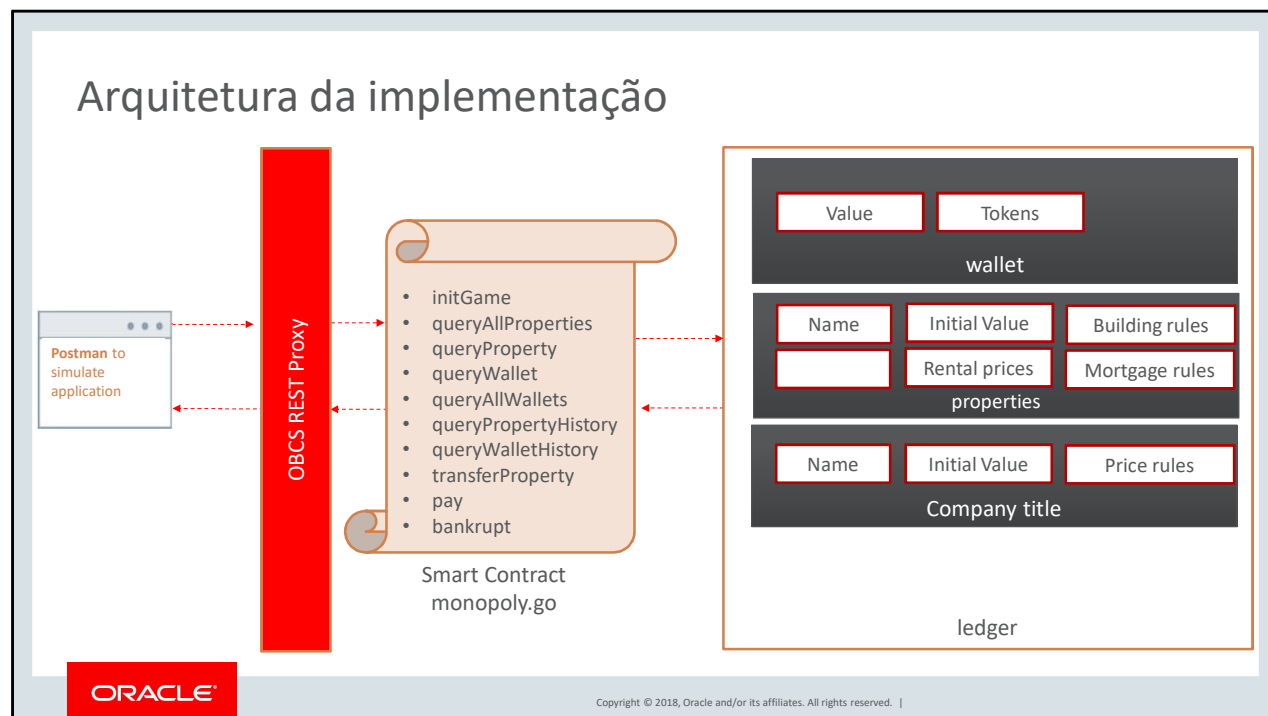
Você precisará criar uma conta ou acessar com sua conta Gmail e está pronto para usar.



Execução das funcionalidades

Para execução das requisições no Postman, especificar:

- Endpoint, conforme instruções a seguir
- Headers: especificar Content-type=application/json
- Especificar Method=POST
- Preencher em Body, com a opção raw, conforme o exemplo de chamada para cada método



Será utilizado o REST Proxy Services provido pelo Oracle Blockchain Cloud Service.

Todas as funcionalidades abaixo poderão ser acionadas a partir de dois endpoints específicos: query e invocation. Os métodos descritos para o endpoint query apenas consultam as informações do estado do ledger. Já os métodos descritos para o endpoint invocation geram uma transação (tx_id) e salvam informações no ledger.

O formato do corpo da mensagem, que pode ser usada tanto para **query** quanto para **invocation**, é:

```
{
  "channel": "<< FILL HERE >>",
  "chaincode": "<< FILL HERE >>",
  "chaincodeVer": "<< FILL HERE >>",
  "method": "<< FILL HERE>>",
  "args": [<< FILL HERE>>]
}
```


Se as instruções de configuração do Oracle Blockchain Cloud Service foram seguidos corretamente neste documento, você deverá usar os parâmetros para as instâncias criadas

para este ambiente.

Conforme instruções ou outras situações que ocorrerem, se necessário, `chaincodeVer` pode assumir valores como `v2`, `v3`, `v4`, etc.


Método	Descrição	Body message
queryAllProperties	Consulta a situação atual de todas as propriedades cadastradas, indicando quem é o proprietário atual.	<pre>{ "channel": "monopolyorderer", "chaincode": "monopoly", "chaincodeVer": "v4", "method": "queryAllProperties", "args": [] }</pre>
queryAllWallets	Consulta a situação atual de todas as carteiras cadastradas, indicando o saldo de cada uma delas.	<pre>{ "channel": "monopolyorderer", "chaincode": "monopoly", "chaincodeVer": "v4", "method": "queryAllWallets", "args": [] }</pre>
queryWallet	Consulta os dados da carteira indicada.	<pre>{ "channel": "monopolyorderer", "chaincode": "monopoly", "chaincodeVer": "v4", "method": "queryWallet", "args": ["Player 1"] }</pre>

Métodos acessíveis através do endpoint QUERY_URL


Copyright © 2018, Oracle and/or its affiliates. All rights reserved. |

Método	Descrição	Body message
queryProperty	Consulta os dados atuais da propriedade indicada.	<pre>{ "channel": "monopolyorderer", "chaincode": "monopoly", "chaincodeVer": "v4", "method": "queryProperty", "args": ["Ipanema"] }</pre>
queryWalletHistory	Consulta o histórico de todas as operações realizadas na carteira especificada.	<pre>{ "channel": "monopolyorderer", "chaincode": "monopoly", "chaincodeVer": "v4", "method": "queryWalletHistory", "args": ["Player 1"] }</pre>
queryPropertyHistory	Consulta o histórico de todas as operações realizadas na propriedade indicada.	<pre>{ "channel": "monopolyorderer", "chaincode": "monopoly", "chaincodeVer": "v4", "method": "queryPropertyHistory", "args": ["Ipanema"] }</pre>


Métodos acessíveis através do endpoint QUERY_URL



Copyright © 2018, Oracle and/or its affiliates. All rights reserved. |

Método	Descrição	Body message
transferProperty	<p>Cria uma transação para transferir uma propriedade do proprietário atual para o novo proprietário, considerando o pagamento da quantia indicada.</p> <p>Para esta versão apenas os terrenos estão sendo considerados como propriedades. As empresas e metrô não.</p>	<pre>{ "channel": "monopolyorderer", "chaincode": "monopoly", "chaincodeVer": "v1", "method": "transferProperty", "args": ["Ipanema", "Banco", "Player 1", 200] }</pre>
Pay	Realiza o pagamento ou transferência de saldos de uma carteira para outra.	<pre>{ "channel": "monopolyorderer", "chaincode": "monopoly", "chaincodeVer": "v1", "method": "pay", "args": ["Player 1", "Banco", 200] }</pre>
bankrupt	Indica que a carteira indicada se encontra em falência. Todo o saldo remanescente será transferido para o Banco.	<pre>{ "channel": "monopolyorderer", "chaincode": "monopoly", "chaincodeVer": "v1", "method": "bankrupt", "args": ["Player 1"] }</pre>


Métodos acessíveis através do endpoint INVOCATION_URL

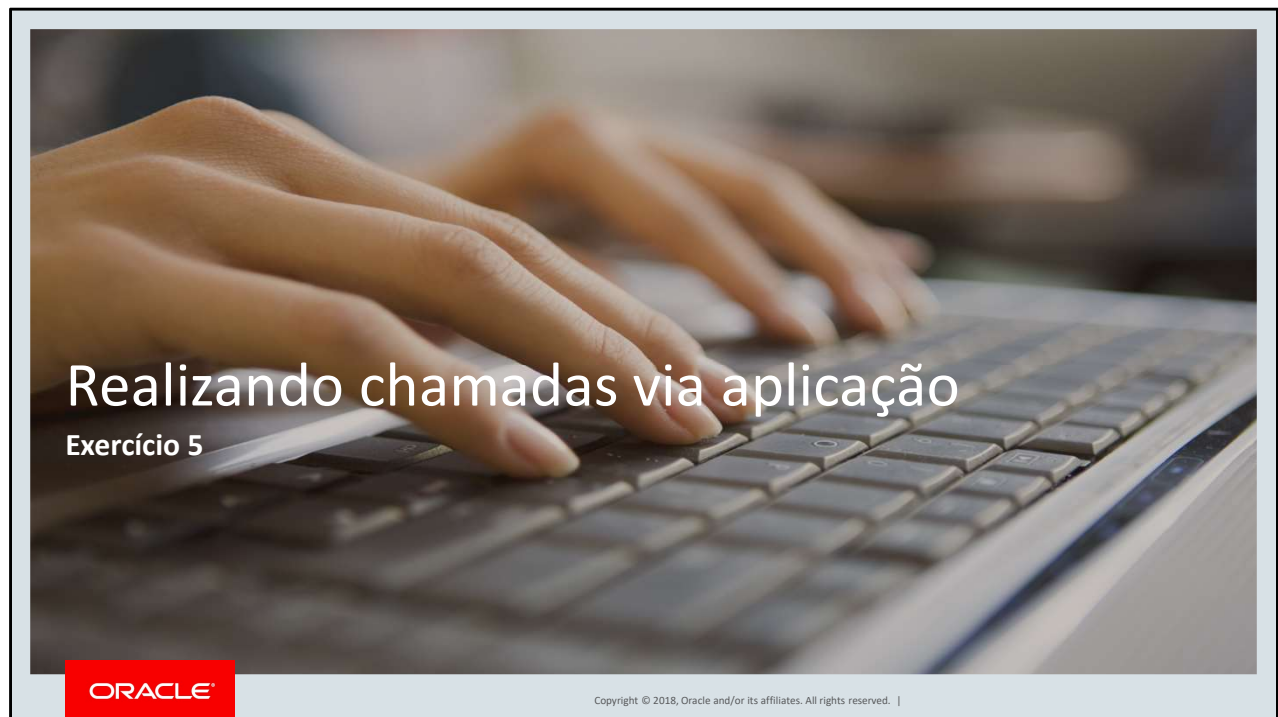


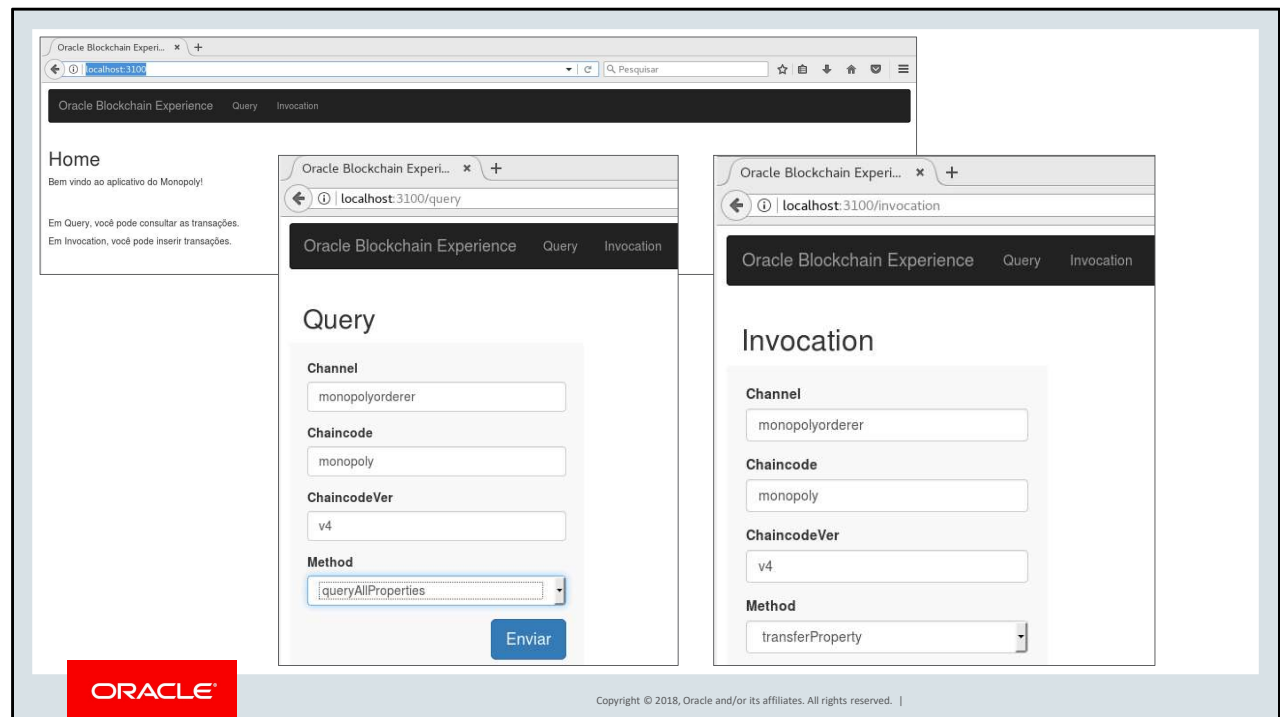
Copyright © 2018, Oracle and/or its affiliates. All rights reserved. |

Método	Descrição	Body message
initGame	<p>Esta transação é chamada automaticamente durante o deployment do contrato inteligente.</p> <p>Porém pode ser acionada a qualquer momento para reinicializar o ambiente:</p> <ul style="list-style-type: none">• Todas as propriedades voltam para o Banco• Todas as carteiras são inicializadas com 1,500 moedas	<pre>{ "channel": "monopolyorderer", "chaincode": "monopoly", "chaincodeVer": "v1", "method": "initGame", "args": [] }</pre>

Métodos acessíveis através do endpoint INVOCATION_URL

 Copyright © 2018, Oracle and/or its affiliates. All rights reserved. |





Ao acessar a URL da aplicação referente ao seu tabuleiro você terá disponível uma aplicação web que interage com o OBCS.

Você terá acesso aos métodos de consulta e aos métodos transacionais disponíveis.

Integrated Cloud

Applications & Platform Services

ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved. |

