Artigo
Danusa Calixto · Dez. 15, 2022 5min de leitura

Entrega contínua de sua solução InterSystems usando GitLab – Parte III: Instalação e configuração do GitLab

Nesta série de artigos, quero apresentar e discutir várias abordagens possíveis para o desenvolvimento de software com tecnologias da InterSystems e do GitLab. Vou cobrir tópicos como:

- Git básico
- Fluxo Git (processo de desenvolvimento)
- Instalação do GitLab
- Fluxo de trabalho do GitLab
- Entrega contínua
- **Instalação e configuração do GitLab**
- CI/CD do GitLab

No primeiro artigo, abordamos os fundamentos do Git, por que um entendimento de alto nível dos conceitos do Git é importante para o desenvolvimento de software moderno e como o Git pode ser usado para desenvolver software.

No segundo artigo, abordamos o fluxo de trabalho do GitLab: um processo inteiro do ciclo de vida do software e a entrega contínua.

Neste artigo, vamos discutir:

- Instalação e configuração do GitLab
- Conexão dos seus ambientes ao GitLab

Instalação do GitLab

Vamos instalar o GitLab no local. Há várias maneiras de instalar o GitLab — da fonte, pacote, em um contêiner. Não descreverei todos os passos aqui, há um guia para isso. Ainda assim, algumas observações.

Pré-requisitos:

- Servidor separado — como é um web application e um recurso bastante intensivo, é melhor executar em um servidor separado
- Linux
- (Opcional, mas altamente recomendável) Domínio — necessário para executar páginas e proteger a configuração inteira

Configuração

Primeiro de tudo, você provavelmente precisa enviar e-mails com notificações.

Em seguida, recomendo instalar Páginas. Como discutido no artigo anterior — artefatos do script podem ser enviados para o GitLab. O usuário pode fazer o download deles, mas é útil poder abri-los diretamente no navegador e, para isso, precisamos de páginas.

Por que você precisa de páginas:

- Para mostrar uma página wiki gerada ou estática para seu projeto
- Para mostrar artefatos html
- Outras coisas que você pode fazer com páginas

Como as páginas html podem ter um redirecionamento onload, elas podem ser usadas para enviar o usuário para onde precisamos. Por exemplo, veja este código que gera uma página html que envia um usuário para o último teste de unidade executado (no momento da geração do html):

```
ClassMethod writeTestHTML()
{
  set text = ##class(%Dictionary.XDataDefinition).IDKEYOpen($classname(), "html").Dat
a.Read()
  set text = $replace(text, "!!!", ..getURL())

  set file = ##class(%Stream.FileCharacter).%New()
  set name = "tests.html"
  do file.LinkToFile(name)
  do file.Write(text)
  quit file.%Save()
}

ClassMethod getURL()
{
  set url = "http://host:57772"
  set url = url _ $system.CSP.GetDefaultApp("%SYS")
  set url = url_"/%25UnitTest.Portal.Indices.cls?Index="_ $g(^UnitTest.Result, 1) _ "
&$NAMESPACE=" _ $zconvert($namespace,"O","URL")
  quit url
}

XData html
{


    If you are not redirected automatically, follow this link to tests.


}
```

Encontrei um bug usando as páginas (erro 502 ao procurar artefatos), veja aqui a correção.

Conexão dos seus ambientes ao GitLab

Para executar scripts de CD, você precisa de ambientes, servidores configurados para executar seu aplicativo. Presumindo que você tem um servidor Linux com o produto InterSystems instalado (digamos InterSystems IRIS, mas funciona também com o Caché e Ensemble), estas etapas conectam o ambiente ao GitLab:

1. Instalar o runner do GitLab
2. Registrar o runner com o GitLab

3. Permitir que o runner chame o InterSystems IRIS

Observação importante sobre a instalação do runner GitLab, NÃO clone servidores após instalar o runner do GitLab. Os resultados são imprevisíveis e muito indesejados.

Registrar o runner com o GitLab

Após executar o inicial:

```
sudo gitlab-runner register
```

você verá vários prompts e, embora a maioria das etapas seja bastante direta, várias não são:

Insira o token gitlab-ci para este runner

Há vários tokens disponíveis:

- Um para o sistema inteiro (disponível nas configurações de administração)
- Um para cada projeto (disponível nas configurações do projeto)

Conforme você conecta um runner para executar a CD para um projeto específico, você precisa especificar um token para este projeto.

Insira as tags gitlab-ci para este runner (separadas por vírgulas):

Na configuração de CD, você pode filtrar quais scripts vão ser executados em quais tags. Então, no caso mais simples, especifique uma tag, que seria o nome do ambiente.

Insira o executor: ssh, docker+machine, docker-ssh+machine, kubernetes, docker, parallels, virtualbox, docker-ssh, shell:
docker

Se você estiver usando o servidor habitual sem docker, escolha shell. O Docker será discutido nas partes posteriores.

Permitir que o runner chame o InterSystems IRIS

Depois de conectar o runner ao GitLab, precisamos permitir que ele interaja com o InterSystems IRIS, para isso:

1. O usuário **gitlab-runner** precisa conseguir chamar csession. Para fazer isso, adicione-o ao grupo cacheusr:
    - `usermod -a -G cacheusr gitlab-runner`
2. Crie o usuário **gitlab-runner** no InterSystems IRIS e dê a ele funções para realizar tarefas de CD (escreva para DB, etc.)
3. Permitir a autenticação no nível do SO

Para 2 e 3, outras abordagens podem ser usadas, como a transmissão de usuário/código, mas acho que a autenticação de SO é preferível.

Conclusão

Nesta parte:

- GitLab instalado

- Ambientes conectados ao GitLab

Links

- Instruções de instalação
- Páginas
- Runner
- Parte I: Git
- Parte II: fluxo de trabalho do GitLab

O que vem a seguir

Na próxima parte, escrevemos nossa configuração de entrega contínua.

#Administração do Sistema #Git #Implantação #Iniciante #Integração Contínua #Caché
