**Customer Relationship Management - CRM Online**

**TDD – Desenvolvimento Dirigido por Teste**

**Versão 1.0**

**Professora:** Maria Augusta Vieira Nelson

**Alunos:** Fernanda Viana Bonfat

Israel de Almeida Mello

Paula Emilia Martins

Rômulo de Oliveira

Túlio Henrique Café

Wander Maia da Silva

Outubro/2012

**Histórico da Revisão**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Versão** | **Descrição** | **Autor** |
| 09/11/2012 | 1.0 | Criação inicial | Rômulo |
| 10/11/2012 | 1.0 | Atualização | Rômulo |
| 11/11/2012 | 1.0 | Atualização | Rômulo |
|  |  |  |  |

**Índice Analítico**

[1. Introdução](http://www.wthreex.com/rup/webtmpl/templates/test/rup_tsteval.htm#1.%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20Introduction)

[2.](http://www.wthreex.com/rup/webtmpl/templates/test/rup_tsteval.htm#2.%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20Test%20Results%20Summary) Testes

[2.1](http://www.wthreex.com/rup/webtmpl/templates/test/rup_tsteval.htm#1.1%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20Purpose) Estrutura Inicial

[2.2](http://www.wthreex.com/rup/webtmpl/templates/test/rup_tsteval.htm#1.1%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20Purpose) Teste 1

[2.3](http://www.wthreex.com/rup/webtmpl/templates/test/rup_tsteval.htm#1.2%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20Scope) Teste 2

[2.4](http://www.wthreex.com/rup/webtmpl/templates/test/rup_tsteval.htm#1.3%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20Definitions,%20Acronyms%20and%20Abbreviations) Teste 3

**Sumário de Avaliação de Testes**

**1.**                  **Introdução**

A rotina de envio de e-mails de lembretes do CRM foi implementada utilizando desenvolvimento dirigido por teste.

A partir dos lembretes configurados pelo usuário o sistema deve enviar um e-mail. Para essa funcionalidade, foi desenvolvido um serviço que executa a ação de notificação por e-mail a cada intervalo de tempo.

O tópico 2 mostra como o usuário pode realizar as configurações para receber e-mails de lembretes.

**2.**                  **Testes**

**1.1**               **Estrutura Inicial**

A estrutura inicial da classe de testes conta possui o estereótipo *TestInitialize* que é executada sempre antes de cada teste. Também possui duas variáveis que são responsáveis pela configuração dos lembretes. Abaixo está a classe:

[TestClass]

public class LembreteTest

{

LembreteEntity lembrete;

LembreteDAO lembreteDAO;

[TestInitialize]

public void TestInitialize1()

{

lembrete = new LembreteEntity();

lembreteDAO = new LembreteDAO();

}

}

}

**1.2**               **Teste 1**

O usuário pode configurar um lembrete em um dia e horário específicos.

* Exemplo: Data da Atividade: 25/12/2012;
* Receber Lembrete nos dias: 01/12/2012 12:00 e 24/12/2012 10:00;

Teste:

[TestMethod]

public void TestMethod1()

{

lembrete.datLem = "01/12/2012";

lembrete.horLem = "12:00";

Assert.IsTrue(lembreteDAO.Verificar(lembrete));

}

O teste falha pois o método não está implementado.

Implementação:

public bool Verificar(LembreteEntity lembrete)

{

DateTime dataAtual = Convert.ToDateTime(DateTime.Now.Date.ToShortDateString());

TimeSpan horaAtual = Convert.ToDateTime(DateTime.Now.TimeOfDay.ToString().Substring(0, 5)).TimeOfDay;

DateTime datLem = Convert.ToDateTime(lembrete.datLem);

TimeSpan horLem = Convert.ToDateTime(lembrete.horLem).TimeOfDay;

if (datLem == dataAtual && horLem == horaAtual)

return true;

else

return false;

}

O teste passa somente na data e na hora especificadas.

**1.3**               **Teste 2**

O usuário pode determinar se deseja receber um lembrete diário, *n* dias antes da data da atividade.

* Exemplo: Data da Atividade: 01/12/2012;
* Receber Lembrete a partir de 3 dias antes da data da atividade às 11:00h; Nesse caso será enviado um e-mail nos dias 28/11/2012, 29/11/2012 e 30/11/2012 às 11:00h;

Teste:

[TestMethod]

public void TestMethod2()

{

lembrete.datAti = "01/12/2012";

lembrete.horAti = "11:00";

lembrete.diaLem = 3;

Assert.IsTrue(lembreteDAO.Verificar(lembrete));

}

O teste falha pois o método não possui implementado este tipo de verificação.

Implementação:

public bool Verificar(LembreteEntity lembrete)

{

DateTime dataAtual = Convert.ToDateTime(DateTime.Now.Date.ToShortDateString());

TimeSpan horaAtual = Convert.ToDateTime(DateTime.Now.TimeOfDay.ToString().Substring(0, 5)).TimeOfDay;

int diaLem = lembrete.diaLem;

if (diaLem == 0)

{

DateTime datLem = Convert.ToDateTime(lembrete.datLem);

TimeSpan horLem = Convert.ToDateTime(lembrete.horLem).TimeOfDay;

if (datLem == dataAtual && horLem == horaAtual)

return true;

}

else

{

DateTime datAti = Convert.ToDateTime(lembrete.datAti);

TimeSpan horAti = Convert.ToDateTime(lembrete.horAti).TimeOfDay;

DateTime datLem = datAti.AddDays(diaLem \* -1);

TimeSpan horLem = horAti;

if (datLem <= dataAtual && datAti > dataAtual && horLem == horaAtual)

return true;

}

return false;

}

O teste passa somente nos dias e na hora dentro do intervalo especificado.

**1.4**               **Teste 3**

O usuário pode configurar se deseja receber e-mails todos os dias da semana ou apenas de segunda a sexta-feira. Caso o dia de envio da mensagem seja um sábado ou domingo e o usuário definiu que não deseja receber e-mail no fim de semana, o envio da mensagem deve ser adiantada.

* Exemplo: Data da Atividade: 01/10/2012;
* Receber Lembrete a partir de 2 dias antes da data a atividade às 11:00h;
* O usuário configurou para não receber e-mails no fim de semana. Como os dias 29/09/2012 e 30/09/2012 são no fim de semana o usuário deve receber um e-mail nos dias 27/09/2012 e 28/09/2012.

Teste:

[TestMethod]

public void TestMethod3()

{

lembrete.datAti = "01/10/2012";

lembrete.horAti = "11:00";

lembrete.diaLem = 2;

lembrete.semLem = true;

Assert.IsTrue(lembreteDAO.Verificar(lembrete));

}

O teste passa, mas por ser sábado não deveria passar. O método ainda não possui implementado este tipo de verificação.

Implementação:

public bool Verificar(LembreteEntity lembrete)

{

DateTime dataAtual = Convert.ToDateTime(DateTime.Now.Date.ToShortDateString());

TimeSpan horaAtual = Convert.ToDateTime(DateTime.Now.TimeOfDay.ToString().Substring(0, 5)).TimeOfDay;

int diaLem = lembrete.diaLem;

if (diaLem == 0)

{

DateTime datLem = Convert.ToDateTime(lembrete.datLem);

TimeSpan horLem = Convert.ToDateTime(lembrete.horLem).TimeOfDay;

if (datLem == dataAtual && horLem == horaAtual)

return true;

}

else

{

DateTime datAti = Convert.ToDateTime(lembrete.datAti);

TimeSpan horAti = Convert.ToDateTime(lembrete.horAti).TimeOfDay;

if (lembrete.semLem == false)

{

DateTime datLem = datAti.AddDays(diaLem \* -1);

TimeSpan horLem = horAti;

if (datLem <= dataAtual && datAti > dataAtual && horLem == horaAtual)

return true;

}

else

{

int fimDeSemana = 0;

for (int i = 1; i <= diaLem; i++)

{

if (datAti.AddDays(i \* -1).DayOfWeek == DayOfWeek.Saturday || datAti.AddDays(i \* -1).DayOfWeek == DayOfWeek.Sunday)

fimDeSemana++;

}

DateTime datLem = datAti.AddDays((diaLem + fimDeSemana) \* -1);

TimeSpan horLem = horAti;

if (dataAtual.DayOfWeek != DayOfWeek.Saturday && dataAtual.DayOfWeek != DayOfWeek.Sunday && datLem <= dataAtual && datAti > dataAtual && horLem == horaAtual)

return true;

}

}

return false;

}

O teste passa somente nos dias de semana e na hora dentro do intervalo especificado.