**JTextField aceitando apenas letras**

**Veja neste artigo como utilizar o evento Key Typed do JTextField para que este aceite apenas letras. Em seguida é apresentado o código de uma classe herdada do JTextField para implementar funcionalidades adicionais.**

 1

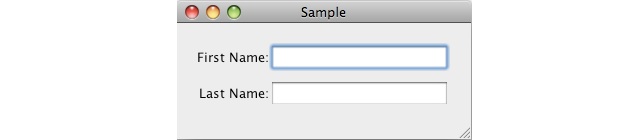
http://www.devmedia.com.br/imagens/portal2010/handUp.png

[Gostei (1)](http://www.devmedia.com.br/jtextfield-aceitando-apenas-letras/26019)

[http://www.devmedia.com.br/articles/img/handDown.png](http://www.devmedia.com.br/jtextfield-aceitando-apenas-letras/26019)

[(0)](http://www.devmedia.com.br/jtextfield-aceitando-apenas-letras/26019)

Neste artigo veremos algumas soluções práticas e de simples implementação para fazer com que um campo de texto aceite apenas letras. Para isso após inserir o JTextField no formulário é necessário alterar o nome do componente, para que quando for necessário acionar as suas propriedades tenhamos uma maior facilidade.



**Figura 1:** JTextFields

Existe um componente que é o JFormattedTextField que é um campo de texto que é utilizado para definirmos a máscara do campo e assim determinarmos como as informações devem ser digitadas.

Não é o nosso objetivo trabalhar nesse post sobre o campo JFormattedTextField e sim operações que podemos aplicar sobre o JTextField para manipulá-lo de modo a exibir as informações com as quais estamos trabalhando, também de forma personalizada.

Vamos ao código para aceitar apenas letras, de forma mais geral podemos apenas criar o evento Key Typed para o componente JTextField e acrescentar o seguinte código dentro do evento Key Typed.

O evento Key Typed: pelo observado durante os testes em que foi desenvolvido esse código é gerado cada vez que um novo caracter tiver de ser adicionado a string, diferente do key pressed e do key relesead, sendo o pressed adicionado quando a tecla é pressionada e o key relesead quando a tecla é solta, ambos não gerando eventos enquanto a tecla permanece pressionada. No nosso caso causando erro durante a execução do algoritmo, por este motivo estamos trabalhando com o key Typed.

**Listagem 1:** Evento Key Tyoed do JTextField para aceitar apenas letras

String caracteres="0987654321";

if(caracteres.contains(evt.getKeyChar()+"")){

evt.consume();

}

Neste código, na variável String, estamos determinando quais caracteres não devem ser aceitos. Neste caso estamos apenas eliminando do componente os números.

E no if verificamos se o caracter que gerou o evento está na variável caracteres, caso esteja acionamos a propriedade consume do evento.

A propriedade consume, pode ser interpretada como:

Consome este evento de modo que não será processada na maneira padrão pela fonte que a originou.

Traduzindo, este evento não deve ser executado.

Para aprimorarmos e conseguirmos reutilizar o código, podemos criar uma classe que herde as características de um JTextField normal, e definir essas propriedades, além disso, podemos também definir o tamanho máximo que o campo deve suportar, dessa maneira também trabalhando a nivel de validação do tamanho da string que deve ser aceita em um JTextField.

**Listagem 2:** Classe herdada de JTextField

public final class JtextFieldSomenteLetras extends JTextField {

private int maximoCaracteres=-1;// definição de -1

// como valor normal de um textfield sem limite de caracters

public JtextFieldSomenteLetras() {

super();

addKeyListener(new java.awt.event.KeyAdapter() {

@Override

public void keyTyped(java.awt.event.KeyEvent evt) {

jTextFieldKeyTyped(evt);}});

}

public JtextFieldSomenteLetras(int maximo) {

super();

setMaximoCaracteres(maximo);// define o tamanho máximo

//que deve ser aceito no jtextfield que foi recebido no construtor

addKeyListener(new java.awt.event.KeyAdapter() {

@Override

public void keyTyped(java.awt.event.KeyEvent evt) {

jTextFieldKeyTyped(evt);}});

}

private void jTextFieldKeyTyped(KeyEvent evt) {

String caracteres="0987654321";// lista de caracters que não devem ser aceitos

if(caracteres.contains(evt.getKeyChar()+"")){// se o character que gerou o evento estiver na lista

evt.consume();//aciona esse propriedade para eliminar a ação do evento

}

if((getText().length()>=getMaximoCaracteres())&&(getMaximoCaracteres()!=-1)){

//if para saber se precisa verificar também o tamanho da string do campo

// maior ou igual ao tamanho máximo, cancela e nao deixa inserir mais

evt.consume();

setText(getText().substring(0,getMaximoCaracteres()));

// esta linha acima é para remover os caracters inválidos caso o usuário tenha copiado o //conteúdo de algum lugar ou seja com tamanho maior que o definido

}//fim do if do tamanho sa string do campo

}

public int getMaximoCaracteres() {

return maximoCaracteres;

}

public void setMaximoCaracteres(int maximoCaracteres) {

this.maximoCaracteres = maximoCaracteres;

}

}

Ao criar essa classe, optamos por já definirmos no construtor a definição de duas opções: a primeira sem parâmetro, que apenas elimina os números do componente e a segunda que recebe como parâmetro o tamanho máximo da string do JTextField e verifica e deixa esses caracteres serem inseridos somente enquanto o tamanho for menor que o informado.

Para utilizar esta classe:

Após criar a classe, e colocar os imports necessários temos duas opções que vamos explicar abaixo.

No NetBeans, basta inserir um JTextField normal, clicar com o botão direito sobre o componente e ir em personalizar código.

Vai existir no lado esquerdo da janela, um combobox com duas opções código padrão e código personalizado. Vamos trocar para código personalizado.

Após a definição de new vamos mudar a classe que está sendo instanciada.

Estava assim:

**Listagem 3:** Código original do JTextField

jTextFieldNome = new javax.swing.JTextField();

Vamos Deixar assim:

**Listagem 4:** Novo código do JTextField

jTextFieldNome = new JtextFieldSomenteLetras();

Neste caso apenas acionamos a opção que vai eliminar os números do teclado.

Ou podemos deixar assim:

**Listagem 5:** Código original do JTextField

jTextFieldNome = new JtextFieldSomenteLetras(50);

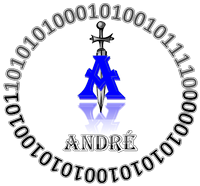
Neste caso acionamos a opção que vai eliminar os números do teclado e também definimos o tamanho máximo do campo em 50 caracteres.

Também podemos instanciar normal caso não esteja utilizando o NetBeans. Basta ir no local que está instanciando a trocar o nome da classe para o JtextFieldSomenteLetras.

Dessa maneira, basta repetir esse processo para todos os campos que pretende eliminar a digitação de números e definir também o tamanho máximo do campo. Facilitando assim o processo de validação dos campos do formulário.

Seguindo o mesmo raciocínio, é possível alterar o padrão de código para criar uma classe que aceite apenas números ou um conjunto de caracteres definidos por nós, já que neste caso trabalhamos eliminando os caracteres da lista.

Até a próxima.

[](http://www.devmedia.com.br/autor/andre-camargo/272384)  
[André Camargo](http://www.devmedia.com.br/autor/andre-camargo/272384)

Analista de Tecnologia da Informaçao:  Bacharel em Ciência da Computação pela URI - Campus Santiago.

O que você achou deste post?

[http://www.devmedia.com.br/imagens/portal2010/handUp.png](http://www.devmedia.com.br/jtextfield-aceitando-apenas-letras/26019)

[Gostei (1)](http://www.devmedia.com.br/jtextfield-aceitando-apenas-letras/26019)

[http://www.devmedia.com.br/articles/img/handDown.png](http://www.devmedia.com.br/jtextfield-aceitando-apenas-letras/26019)

[(0)](http://www.devmedia.com.br/jtextfield-aceitando-apenas-letras/26019)

[Comentário | Tire sua dúvida](http://www.devmedia.com.br/rte3/addcomment.asp?idcomp=26019)



Felipe Mendes Silva

bom mesmo heim cara, me ajudou.

[há +1 ano] - [Responder](http://www.devmedia.com.br/rte3/addcomment.asp?idcomp=26019&idco_pai=41416)

Conhece a assinatura MVP?

Serviços

[Inclua um comentário](http://www.devmedia.com.br/rte3/addcomment.asp?idcomp=26019)

[Adicionar aos Favoritos](http://www.devmedia.com.br/favorite/addfavorite.asp?comp=26019)

[Marcar como lido/assistido](http://www.devmedia.com.br/favorite/add_finished.asp?comp=26019)

[Incluir anotação pessoal](http://www.devmedia.com.br/anotacoes/add/?idcomp=26019)

[Versão para impressão](http://www.devmedia.com.br/articles/viewcomp_forprint.asp?comp=26019)

[+Java](http://www.devmedia.com.br/java)

Mais posts

* Artigo

[Serializando objetos com byte em Java e PostgreSQL](http://www.devmedia.com.br/serializando-objetos-com-byte-em-java-e-postgresql/32089)

* Artigo

[T-SQL Subqueries: Serializando objetos em Java e PostgreSQL](http://www.devmedia.com.br/t-sql-subqueries-serializando-objetos-em-java-e-postgresql/32072)

* Pocket Video

[Integrando o DropBox com o Java](http://www.devmedia.com.br/integrando-o-dropbox-com-o-java/31961)

* Revista

[Revista easy Java Magazine 47](http://www.devmedia.com.br/revista-easy-java-magazine-47/32020)

* Artigo

[Programação paralela em Java](http://www.devmedia.com.br/programacao-paralela-em-java/32010)

* Artigo

[Criando aplicações Desktop em Java](http://www.devmedia.com.br/criando-aplicacoes-desktop-em-java/32009)

* Artigo

[Anti-padrões de Projeto: o que são, como identificar e evitar](http://www.devmedia.com.br/anti-padroes-de-projeto-o-que-sao-como-identificar-e-evitar/32008)

* Video aula

[Implementando o filtro - Curso de Java Web: Servlet, JSP, JSTL e Tags - Aula 100](http://www.devmedia.com.br/implementando-o-filtro-curso-de-java-web-servlet-jsp-jstl-e-tags-aula-100/31933)

* Video aula

[SQL de filtragem - Curso de Java Web: Servlet, JSP, JSTL e Tags - Aula 99](http://www.devmedia.com.br/sql-de-filtragem-curso-de-java-web-servlet-jsp-jstl-e-tags-aula-99/31932)

[Listar mais conteúdo](http://www.devmedia.com.br/java)

Leia mais em: [JTextField aceitando apenas letras](http://www.devmedia.com.br/jtextfield-aceitando-apenas-letras/26019#ixzz3Qh7S3zCl) <http://www.devmedia.com.br/jtextfield-aceitando-apenas-letras/26019#ixzz3Qh7S3zCl>