**O que é locale?**

Java é uma linguagem que dá suporte à internacionalização de suas aplicações, oferecendo diversos recursos que facilitam o desenvolvimento de programas cujas interfaces (sejam elas CLIs ou GUIs) se adaptam a cada região.

Para fazer isto, ao ser iniciada, a JVM elege um conjunto de configurações baseadas no sistema em que ela está sendo executada, chamado locale, que define uma linguagem, uma região e quaisquer variações na maneira de exibir e interpretar certos tipos de dados.

Esses parâmetros ficam armazenados em um objeto Locale que possui um Enum Category com duas constantes, DISPLAY e FORMAT, que armazenam [códigos de línguas,](https://en.wikipedia.org/wiki/Language_code)que são consultadas por métodos de outras classes para determinar o tratamento de alguns tipos de dados. Esses métodos pertencem a objetos **sensíveis ao locale**.

Podemos descobrir o locale padrão da nossa JVM com o seguinte código:

import java.util.Locale;

public class TesteLocale {

public static void main(String[] args) {

System.out.println(Locale.getDefault(Locale.Category.DISPLAY));

System.out.println(Locale.getDefault(Locale.Category.FORMAT));

}

}

Como exemplo de um programa que possui objetos sensíveis ao locale, vamos imaginar um programa que calcula o perímetro de uma circunferência e mostra o resultado:

import java.util.Scanner;

public class ProgramaPerímetro {

public static void main(String[] args) {

final double RAIO;

try(Scanner entrada = new Scanner(System.in)) {

System.out.print("Digite o raio: ");

RAIO = entrada.nextDouble();

}

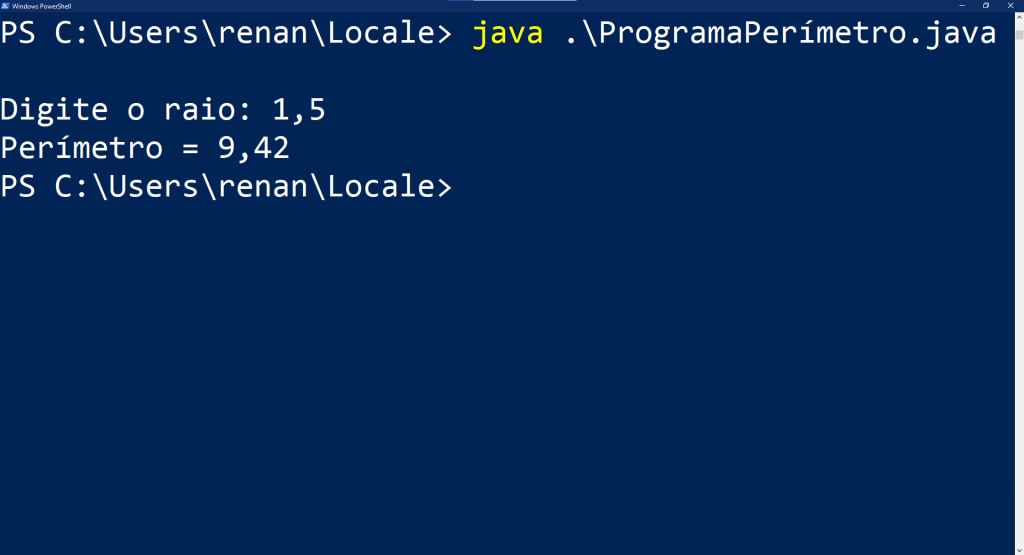
System.out.format("Perímetro = %.2f%n", 2\*Math.PI\*RAIO);

}

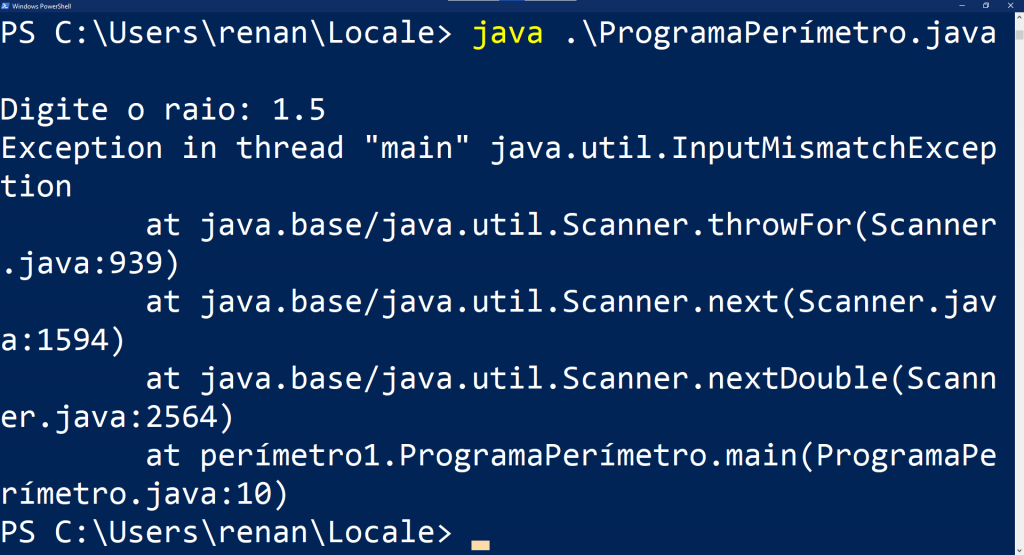
}

Nele existem dois objetos sensíveis ao locale: System e Scanner. Os seus métodos, format e nextDouble, respectivamente, consultam a constante FORMAT do Locale afim de determinar a formatação dos números double, que serão exibidos ou recebidos.

Ao iniciar a JVM no meu sistema, ela atribui à constante FORMAT o código pt\_BR, fazendo com que os métodos que lidam com números double imprimam e esperem receber números com uma vírgula como separador decimal.



Caso eu digite o valor do raio usando ponto como separador decimal (o separador padrão nos Estados Unidos), o nextDouble disparará uma exceção InputMismatchException, já que ele esperava receber um número com uma vírgula como separador decimal.



Se quisermos que os objetos sensíveis ao locale utilizem outros parâmetros além dos padrões, podemos instanciar um novo objeto locale da seguinte forma:

import java.util.Locale;

import java.util.Scanner;

public class ProgramaPerímetro {

public static void main(String[] args) {

final double RAIO;

try(Scanner entrada = new Scanner(System.in)) {

System.out.print("Digite o raio: ");

// criação do novo objeto Locale

final var localeAmericano = new Locale("en", "US");

// nextDouble usará o Locale criado graças ao método useLocale

RAIO = entrada.useLocale(localeAmericano).nextDouble();

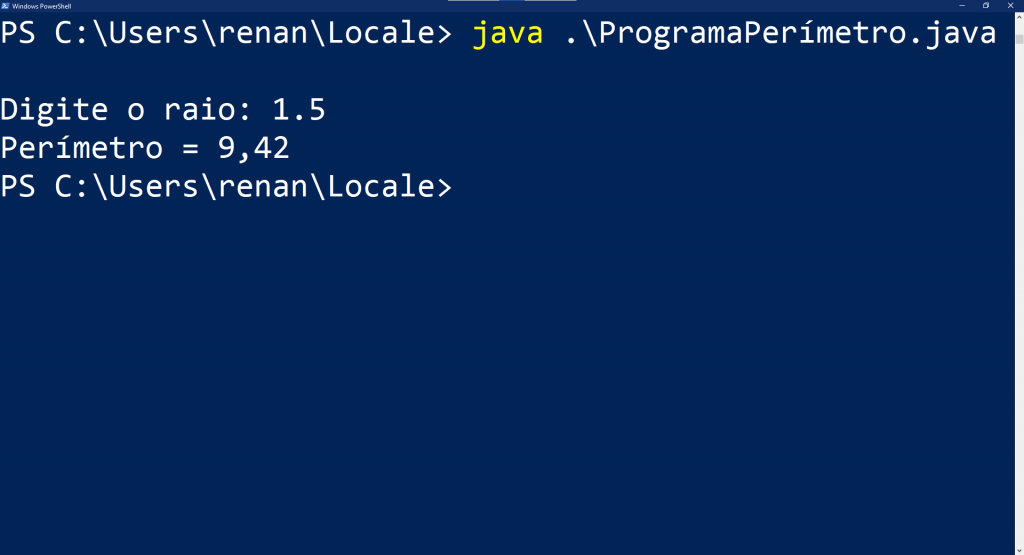
}

System.out.format("Perímetro = %.2f%n", 2\*Math.PI\*RAIO);

}

}

No programa acima, o método format continua exibindo os números com uma vírgula como separador decimal, como é padrão no Brasil, porém espera receber um número com o padrão de formatação americano. Isso ocorre graças ao novo objeto Locale, que instanciamos com os argumentos “en” e “US”, que formam o código de língua e de região americano, e o método useLocale.



Como alternativa, podemos determinar um novo locale padrão no início do programa:

import java.util.Locale;

import java.util.Scanner;

public class ProgramaPerímetro {

public static void main(String[] args) {

// criação e definição do novo Locale padrão

final var localeAmericano = new Locale("en", "US");

Locale.setDefault(localeAmericano);

final double RAIO;

try(Scanner entrada = new Scanner(System.in)) {

System.out.print("Digite o raio: ");

RAIO = entrada.nextDouble();

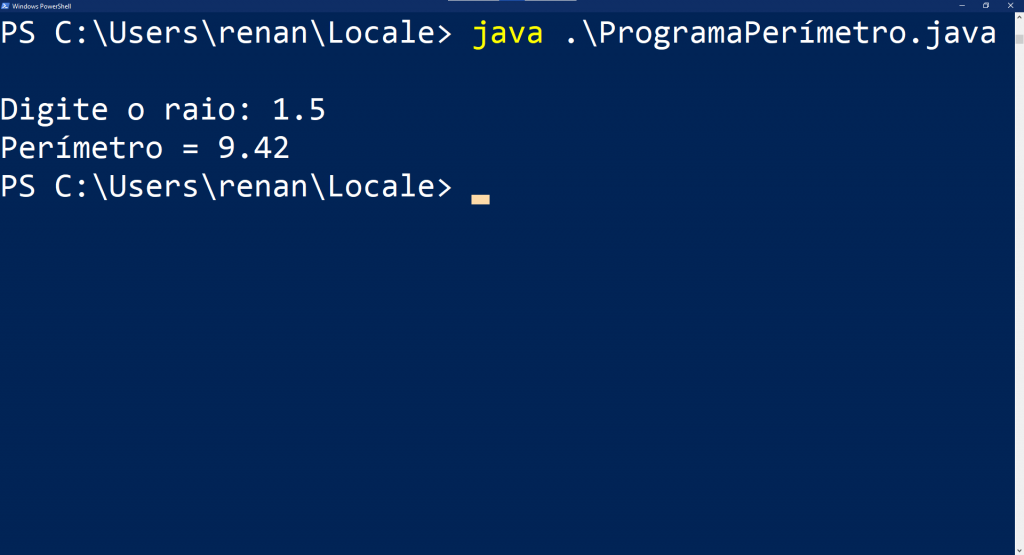
}

System.out.format("Perímetro = %.2f%n", 2\*Math.PI\*RAIO);

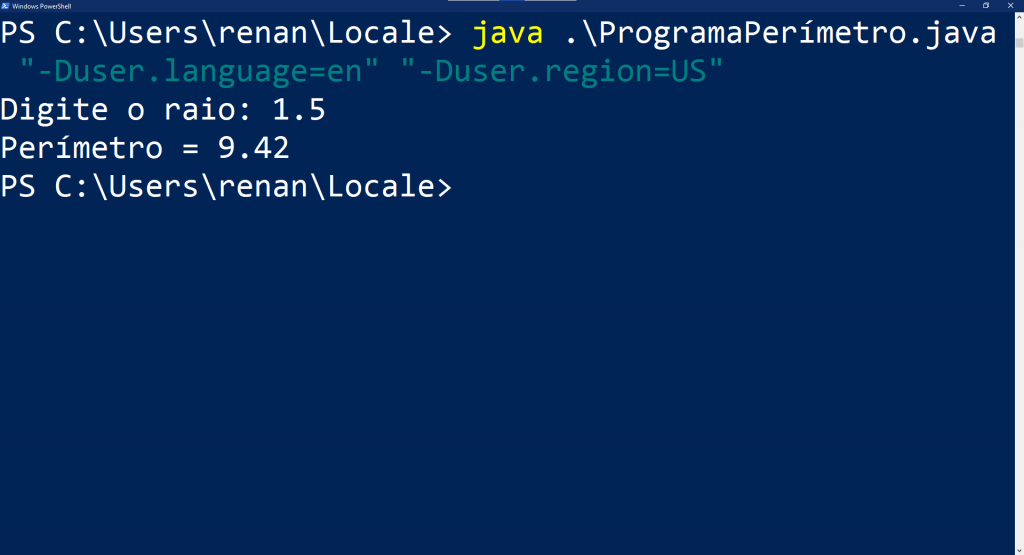
}

}

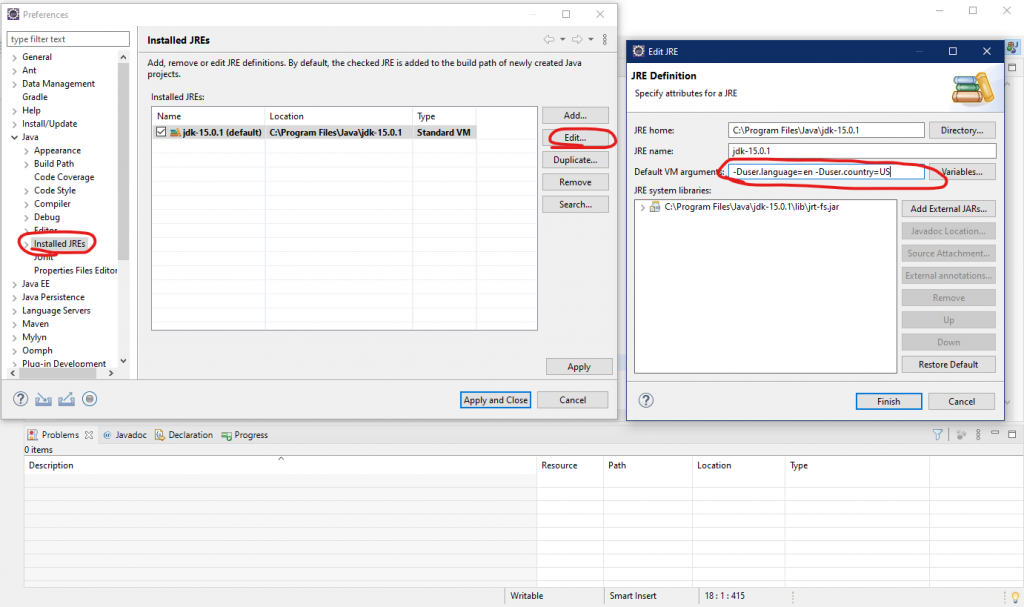
Dessa forma, todos os métodos de objetos sensíveis ao locale agora formatam os dados de acordo com os padrões americanos.



O mesmo resultado pode ser alcançado usando os argumentos -Duser.language=en -Duser.country=US ao lançar a JVM, que força ela a usar o locale passado como argumento como padrão. Além disso, evita que instanciemos um objeto Locale no corpo do programa.



No Eclipse, podemos fazer acessando a tela em *Window*>*Java*>*Installed JREs*, selecionando a instalação do Java para a qual você quer estabelecer um locale padrão, clicando em *Edit…*, e pondo os argumentos para o locale desejado no campo *Default VM arguments*. Como mostra a imagem abaixo:



O objeto Locale faz parte dos recursos que o Java nos disponibiliza para a criação de aplicações localizadas. Entretanto é muito importante ficar atento a qual locale sua JVM está usando como padrão para evitar *bugs* relacionados a má interpretação de dados.

Espero ter esclarecido esse importante conceito do universo Java. Bons estudos!