Objetivos:

- Criar camada no QGIS utilizando arquivo CSV (Comma-Separated Values, em português, Valores Separados por Vírgula);
- Escrever arquivo CSV a partir de dados contidos em arquivos Shapefile ou GeoPackage.

I. Arquivos CSV

Os arquivos CSV (Comma-Separated Values, ou Valores Separados por Vírgula) são um dos formatos mais comuns para o armazenamento e a troca de dados tabulares. Sua simplicidade e ampla compatibilidade com diversos programas fazem do CSV uma escolha frequente para representar tabelas de dados em ambientes de SIG, como o QGIS.

Em um arquivo CSV:

- Cada **linha** representa um **registro**;
- Cada **campo** de um registro é separado por um **delimitador**, normalmente uma vírgula (,). Contudo, é frequente o uso de outros delimitadores, como o **ponto e vírgula** (;), especialmente em países onde a vírgula é usada como separador decimal, assim como no Brasil;
- Arquivos CSV são arquivos de **texto plano** e podem ser abertos e editados com editores de texto simples, como o Bloco de Notas, ou com softwares especializados, como planilhas eletrônicas (Microsoft Excel, Google Sheets, entre outros).

Exemplo de arquivo CSV:

```
nome,longitude,latitude,populacao
Jacareí,-45.96642,-23.30555,240275
Jambeiro,-45.69457,-23.25871,6828
Santa Branca,-45.88441,-23.39755,13975
```

Nesse exemplo:

- A primeira linha contém os nomes dos campos: `nome`, `longitude`, `latitude` e `populacao`;
- As linhas subsequentes contêm os dados correspondentes a cada cidade.

Considerações importantes:

- Delimitador correto: ao carregar o arquivo, verifique qual delimitador está sendo usado;
- Codificação de caracteres (encoding): certifique-se de que o arquivo está salvo com uma codificação compatível, como UTF-8, para evitar problemas com acentuação;
- Consistência dos dados: verifique se todas as linhas seguem o mesmo padrão de quantidade e tipo de campos.

II. Leitura de arquivo CSV e criação de camada no QGIS

Utilizando a API do PyQGIS, é possível abrir um arquivo CSV e criar dinamicamente uma nova camada vetorial no QGIS.

Passos Metodológicos:

1. **Importação da biblioteca para manipular CSV**

O Python possui uma biblioteca padrão para manipulação de arquivos CSV chamada `csv`. A função `csv.reader()` permite ler arquivos delimitados por vírgulas ou outros caracteres.

```
import csv
```

2. **Abertura do arquivo CSV**

O arquivo CSV é aberto em modo de leitura (`"r"`), definindo a codificação como `"utf-8"` para garantir compatibilidade com acentuação e caracteres especiais.

```
entrada = "D:/pasta/cidades.csv"
arquivo = open(entrada, mode="r", newline="", encoding="utf-8")
```

3. **Leitura das linhas do arquivo CSV**

Utiliza-se o método `csv.reader()` informando o delimitador usado no arquivo. Neste exemplo, a vírgula `","` é utilizada.

```
leitor = csv.reader(arquivo, delimiter=",")
```

4. **Criação da camada de saída**

Cria-se uma camada vetorial temporária, na memória, com tipo de geometria `Point`, sistema de referência de coordenadas `EPSG:4326`, e com dois campos de atributos: `nome` (string) e `populacao` (inteiro).

```
camada =
QgsVectorLayer('Point?crs=EPSG:4326&field=nome:string&field=populacao:integer',
                        'exemplo1',
                        'memory')
```

5. **Obtenção do provedor de dados e campos**

O provedor de dados (`dataProvider()`) é responsável por permitir a inserção e modificação de feições na camada. Também são obtidos os campos definidos.

```
provider = camada.dataProvider()
campos = camada.fields()
```

6. **Processamento das linhas do CSV e criação das feições**

Percorre-se cada linha do arquivo. A **primeira linha** é o cabeçalho e, por isso, é ignorada utilizando uma condição que verifica se o conteúdo da primeira coluna (`linha[0]`) é diferente de `"nome"`.

```
for linha in leitor:
```

```
    if linha[0] != "nome":
        # Cria o ponto com coordenadas X (longitude) e Y (latitude)
        ponto = QgsPointXY(float(linha[1]), float(linha[2]))

        # Converte o ponto para uma geometria
        ponto = QgsGeometry().fromPointXY(ponto)

        # Cria uma feição associada aos campos da camada
        feicao = QgsFeature(campos)

        # Define a geometria da feição
        feicao.setGeometry(ponto)

        # Define os atributos da feição
        feicao.setAttribute("nome", linha[0])  # o nome está na coluna 0
        feicao.setAttribute("populacao", int(linha[3]))  # a população está na coluna 3

        # Adiciona a feição ao provedor de dados
        provider.addFeature(feicao)
```

7. Encerramento do arquivo CSV

Após a leitura completa, o arquivo é fechado.

```
arquivo.close()
```

8. Adição da camada ao projeto do QGIS

Por fim, a nova camada criada é adicionada ao painel de camadas do QGIS, para visualização e processamento.

```
QgsProject().instance().addMapLayer(camada)
```

Exemplo 1: Lê as linhas do arquivo cidades.csv e cria a camada denominada `exemplo1` no QGIS.

Observação: é necessário colocar o caminho correto do arquivo em `"D:/pasta/cidades.csv"`.

```
import csv

entrada = "D:/pasta/cidades.csv"
# Abre o arquivo para leitura
arquivo = open(entrada, mode="r", newline="", encoding="utf-8")
leitor = csv.reader(arquivo, delimiter=",")

# Cria uma camada temporária do tipo Point com feições com os campos nome e populacao
camada = QgsVectorLayer('Point?crs=EPSG:4326&field=nome:string&field=populacao:integer',
'exemplo1', 'memory')
# Obtém o provedor de dados
provider = camada.dataProvider()
campos = camada.fields() # Obtém os campos das feições

for linha in leitor:
```

```
    if linha[0] != 'nome':
        # Cria o ponto com coordenadas X (longitude) e Y (latitude)
        ponto = QgsPointXY(float(linha[1]), float(linha[2]))

        # Converte o ponto para uma geometria
        ponto = QgsGeometry().fromPointXY(ponto)

        # Cria uma feição associada aos campos da camada
        feicao = QgsFeature(campos)

        # Define a geometria da feição
        feicao.setGeometry(ponto)

        # Define os atributos da feição
        feicao.setAttribute("nome", linha[0])  # o nome está na coluna 0
        feicao.setAttribute("populacao", int(linha[3]))  # a população está na coluna 3

        # Adiciona a feição ao provedor de dados
        provider.addFeature(feicao)

# fechar o arquivo
arquivo.close()

# Adiciona a camada ao projeto atual
QgsProject().instance().addMapLayer(camada)
```

III. Escrevendo Arquivos CSV no QGIS com Python

Após aprender como ler arquivos CSV para criar camadas vetoriais no QGIS, é fundamental compreender o processo inverso: exportar dados geográficos de uma camada para um arquivo CSV.

Esse processo é útil quando se deseja compartilhar informações tabulares, realizar análises em softwares estatísticos ou armazenar resultados de processamento geográfico.

Exemplo prático: Exportando pontos de uma camada para CSV

Neste exemplo, será utilizada a camada vetorial capital.gpkg, que contém pontos representando as capitais dos estados brasileiros. Esta camada possui dois atributos:

- gid: identificador numérico da feição;
- nome: nome da capital.

Além desses atributos, extrairemos as coordenadas geográficas (longitude e latitude) de cada ponto para compor o CSV.

Passos Metodológicos:

1. **Abrir a camada vetorial**

Utiliza-se a classe `QgsVectorLayer` para criar uma camada a partir do arquivo capital.gpkg.

```
entrada = "D:/pasta/capital.gpkg"
# Carrega a camada
camada = QgsVectorLayer(entrada, "capitais", "ogr")
```

2. Abrir o arquivo CSV para escrita

Usa-se a função `open()` no modo `"w"` (write), com codificação `"utf-8"` para garantir a compatibilidade com caracteres especiais.

```
open(saida, mode="w", newline="", encoding="utf-8")
```

O bloco `with` … `as` `arquivo` faz o código de abertura do arquivo estar na variável `arquivo` e ser acessível dentro do bloco.

O bloco `with` … `as` permite que o arquivo seja automaticamente fechado ao final do bloco de instruções.

3. Criar o objeto csv.writer

Esse objeto facilita a escrita de linhas no arquivo CSV, utilizando a vírgula como delimitador.

```
escritor = csv.writer(arquivo, delimiter=",")
```

4. Escrever o cabeçalho do CSV

Antes de escrever os dados, cria-se o cabeçalho com os nomes das colunas: `gid`, `nome`, `longitude` e `latitude`.

```
escritor.writerow(["gid", "nome", "longitude", "latitude"])
```

5. Iterar sobre as feições da camada

Para cada feição:

- Obtém-se os atributos `gid` e `nome`:

```
for feicao in camada.getFeatures():
    gid = feicao["gid"]
    nome = feicao["nome"]
```

- Extrai-se a geometria:

```
geom = feicao.geometry()
ponto = geom.asPoint()
longitude = ponto.x()
latitude = ponto.y()
```

- Escrevem-se os dados no CSV:

```
escritor.writerow([gid, nome, longitude, latitude])
```

6. O arquivo CSV é automaticamente fechado

Graças ao uso do bloco `with`, o arquivo é fechado automaticamente ao final da execução.

```
    arquivo.close()
```

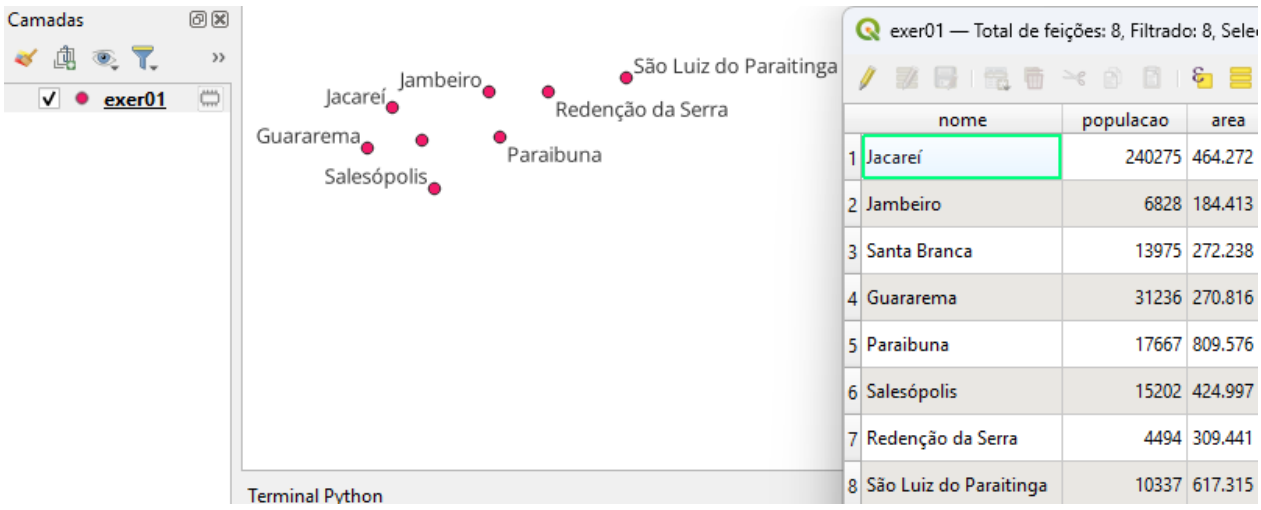
Exemplo 2: Observação: é necessário colocar o caminho correto do arquivo de entrada GPKG e saída CSV.

```
import csv

entrada = "D:/pasta/capital.gpkg"
saida = "D:/pasta/capitais.csv"

camada = QgsVectorLayer(entrada, "capitais", "ogr")

with open(saida, mode="w", newline="", encoding="utf-8") as arquivo:
    escritor = csv.writer(arquivo, delimiter=",")
    escritor.writerow(["gid", "nome", "longitude", "latitude"])

    # Itera sobre as feições da camada
    for feicao in camada.getFeatures():
        gid = feicao["gid"]
        nome = feicao["nome"]
        geom = feicao.geometry()
        ponto = geom.asPoint()
        longitude = ponto.x()
        latitude = ponto.y()

        escritor.writerow([gid, nome, longitude, latitude]) # Escreve a linha no CSV
```

Exercícios

Veja o vídeo se tiver dúvidas nos exercícios: https://youtu.be/JVO5nRfYnYs

Exercício 1: Altere o código do Exemplo 1 para ler o arquivo cidades-exer01.csv e carregar como nova camada do QGIS.



Dicas:

- Use o delimitador ponto e vírgula;
- No construtor `QgsVectorLayer` será necessário incluir o campo área com o tipo de dado float (`&field=area:float`);
- Adicione o atributo `area` em cada feição criada:

```
feicao.setAttribute("area", float(linha[4]))  # a área está na coluna 4
```

Exercício 2: Modifique o código do Exemplo 2 para ler o arquivo uf.gpkg e escrever em um arquivo CSV os atributos gid, uf e as coordenadas do centroide de cada polígono

Dicas:

- O método geom = `feicao.geometry()` retorna uma geometria do tipo Polygon, então precisaremos chamar o método `geom.centroid()` para obter uma geometria com o ponto central;
- Converta a geometria para ponto usando `geom.asPoint()`. Essa conversão é necessária para podermos obter as coordenadas X e Y do ponto;
- Use os métodos `x()` e `y()` do Point para obter as coordenadas.

Parte do arquivo CSV de resposta:

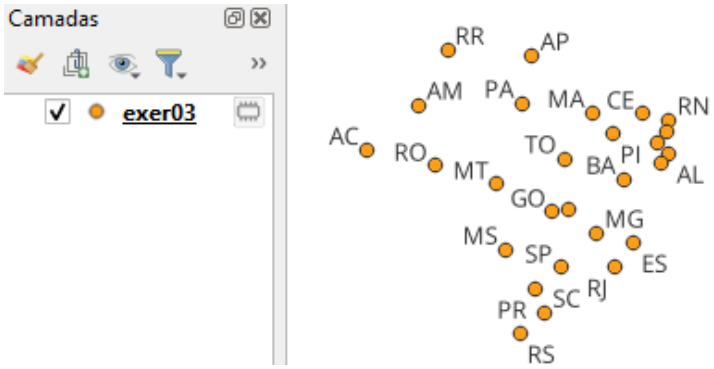
```
gid,uf,longitude,latitude
1,SC,-50.48112567049315,-27.24626307693253
2,RS,-53.320293828693416,-29.705947308226833
3,MG,-44.67339353154309,-18.456243841230318
```

Exercício 3: Modifique o código do Exercício 1 para ler o arquivo CSV criado no Exercício 2 e carregar como camada do QGIS.

Dicas:

- O delimitador é a vírgula;
- A camada possui apenas o campo uf.

Resposta:



Exercício 4: Modifique o código do Exercício 2 para obter o centroide dos municípios do estado de SP que estão no arquivo municipio.gpkg.

Dica:

- No corpo da estrutura de repetição `for`, inclua um `if` para verificar se a feição possui o campo `uf` igual a `"SP"`.

```
# Itera sobre as feições da camada
for feicao in camada.getFeatures():
    if feicao["uf"] == "SP":
```

Parte do arquivo CSV de resposta:

```
gid,municipio,uf,longitude,latitude
589,Rosana,SP,-52.83722842789965,-22.488941154615855
691,Euclides da Cunha Paulista,SP,-52.58930469075374,-22.519071571725316
724,Teodoro Sampaio,SP,-52.37386582570553,-22.416591580448532
```

Exercício 5: Modifique o código do Exercício 3 para ler o arquivo CSV criado no Exercício 4 e carregar como camada do QGIS.

Resposta: