Estruturas de arquivos: estrutarq

Release 0.1

Jander Moreira

04 abr. 2022

Conteúdo:

1 Introdução	1
Índice de Módulos Python	17
Índice	19

CAPÍTULO 1

Introdução

O pacote estrutarq foi moldado para dar suporte ao livro Estruturas de arquivos: uma abordagem prática.

A implementação desenhada é simplificada. Em especial:

- o código é voltado à legibilidade e não ao desempenho
- controle e recuperação de erros são mantidos no nível mínimo, restrito ao âmbito de controle de exceções
- aspectos de acesso simultâneo aos dados são ignorados e, assim, não estão disponíveis mecanismos de exclusão mútua ou escalonamento de acesso

1.1 Conteúdo geral

1.1.1 Pacote estrutarq.dado

Módulo estrutarq.dado_comum

Estruturação de dados para armazenamento interno, gravação e leitura, usando representações diversas:

- Em representação bruta
- · Com terminador
- Prefixada pelo comprimento
- Em formato binário
- De comprimento fixo predefinido

Licença: GNU GENERAL PUBLIC LICENSE V.3, 2007

Jander Moreira, 2022

As classes providas pelo módulo, importadas direramente de estrutarq.dado, são a interface para a criação das organizações de dados disponíveis.

Dado bruto

class estrutarq.dado.DadoBruto

Base: estrutarq.dado.dado_comum.DadoBasico

Classe para dado em forma bruta, ou seja, sem acréscimo de qualquer forma de organização de dados.

Campos brutos não possuem aplicação prática e são usados apenas para fins didáticos.

```
adicione\_formatacao(dado: bytes) \rightarrow bytes
```

Para dado bruto não há acréscimo de bytes de organização de dados e o dado é repassado sem modificação.

Parâmetros dado (*bytes*) – bytes do dado

Retorna bytes do dado inalterados

Tipo de retorno bytes

leia_de_arquivo(arquivo: BinaryIO)

Recuperação de um dado lido de um arquivo (inviável para dado bruto).

Parâmetros arquivo (BinaryIO) – arquivo binário aberto com permissão de leitura

Levanta NotImplemented – se o método for acidentalmente chamado

leia_de_bytes(sequencia: bytes)

Recuperação de um dado extraído de uma sequência de bytes (inviável para dado bruto).

Parâmetros sequencia (bytes) – sequência de bytes

Levanta NotImplemented – se o método for acidentalmente chamado

remova_formatacao(sequencia: bytes) \rightarrow bytes

Para o dado bruto não há bytes de organização a sequência de bytes é repassada sem modificação.

Parâmetros sequencia (*bytes*) – uma sequência de bytes

Retorna a sequência inalterada

Tipo de retorno bytes

Dado com terminador

class estrutarq.dado.DadoTerminador(terminador: bytes)

Base: estrutarq.dado.dado_comum.DadoBasico

Classe para implementação de dados com terminador. O dado é tratado como uma sequência de bytes à qual um byte predefinido (*terminador*) é acrescentado ao final para demarcar o fim dos dados. A existência do valor do byte terminador na sequência de dados é tratada com a técnica de enchimento de bytes (implementada em *DadoBasico*).

Parâmetros terminador (bytes) – um byte a ser usado como terminador

```
adicione\_formatacao(dado: bytes) \rightarrow bytes
```

Formatação do dado: uso de 'byte stuffing' para permitir o byte terminador como dado e acréscimo do byte terminador.

Parâmetros dado (bytes) – sequência de bytes do dado

Retorna a sequência de dados enchida e com o acréscimo do terminador ao final

Tipo de retorno bytes

leia_de_arquivo(arquivo: BinaryIO) \rightarrow bytes

Leitura de um único dado com terminador. A leitura é feita byte a byte até que o byte terminador seja encontrado. Bytes terminadores enchidos são restaurados, mas não determinam o fim da busca. O enchimento de bytes é removido.

Parâmetros arquivo – arquivo binário aberto com permissão de leitura

Retorna a sequência de bytes do dado sem o terminador

Levanta EOFError – se o fim do arquivo for atingido antes de o byte terminador ser encontrado

```
leia_de_bytes(sequencia: bytes) → tuple[bytes, bytes]
```

Recuperação de um dado individual de uma sequência de bytes, retornando o dado até o terminador e o restante da sequência depois do terminador.

Parâmetros sequencia (*bytes*) – uma sequência de bytes

Retorna uma tupla contendo os bytes dos dados e a sequência de bytes restante, excluindo-se de ambas o terminador

Tipo de retorno tuple[bytes, bytes]

```
remova_formatacao(sequencia: bytes) \rightarrow bytes
```

Desformatação do dado: remoção dos bytes de enchimento e também do byte terminador.

Parâmetros sequencia – sequência de bytes de dados

Retorna sequência de bytes de dados, esvaziada e sem terminador

Tipo de retorno bytes

Levanta TypeError – se o terminador não estiver presente na sequência esvaziada

property terminador: bytes

Byte simples usado como terminador.

Dado prefixado pelo comprimento

class estrutarq.dado.DadoPrefixado

Base: estrutarq.dado.dado_comum.DadoBasico

Classe dado prefixados pelo seu comprimento

```
adicione\_formatacao(dado: bytes) \rightarrow bytes
```

Formatação do dado: acréscimo do prefixo binário com comprimento (2 bytes, big-endian, sem sinal).

Parâmetros dado – valor do dado

Retorna o dado formatado

$leia_de_arquivo(arquivo: BinaryIO) \rightarrow bytes$

Leitura de um único dado prefixado pelo comprimento.

Parâmetros arquivo – arquivo binário aberto com permissão de leitura

Retorna os bytes do dado

O comprimento é armazenado como um inteiro de 2 bytes, big-endian. Em caso de falha na leitura é lançada a exceção EOFError

```
leia_de_bytes(sequencia: bytes) -> (<class 'bytes'>, <class 'bytes'>)
```

Recuperação de um dado individual de uma sequência de bytes, retornando o dado sem o prefixo e o restante da sequência.

Parâmetros sequencia – uma sequência de bytes

Retorna os bytes de dados e o restante da sequência

```
remova\_formatacao(sequencia: bytes) \rightarrow bytes
```

Desformatação do dado: remoção dos dois bytes do comprimento.

Parâmetros sequencia – bytes de dados

Retorna dado efetivo, sem o prefixo de comprimento

Dado binário

class estrutarq.dado.DadoBinario(comprimento: int)

Base: estrutarq.dado.dado_comum.DadoBasico

Classe para dados binários com um determinado comprimento em bytes. O comprimento é fixo.

Parâmetros comprimento (int) – comprimento em bytes do valor a ser armazenado

 $adicione_formatacao(dado: bytes) \rightarrow bytes$

Formatação do dado: apenas repassa o dado binário.

Parâmetros dado – valor binário

Retorna o dado formatado

leia_de_arquivo(arquivo: BinaryIO) \rightarrow bytes

Recuperação dos bytes do valor binário a partir de um arquivo.

Parâmetros arquivo – arquivo binário aberto com permissão de leitura

Retorna a sequência de bytes lidos

leia_de_bytes(sequencia: bytes) -> (<class 'bytes'>, <class 'bytes'>)

Recuperação de um dado binário de comprimento definido a partir de uma sequência de bytes.

Parâmetros sequencia – sequência de bytes

Retorna tupla com os bytes do dado, removidos os bytes de organização de dados, e a sequência de bytes restante

remova_formatacao(sequencia: bytes) \rightarrow bytes

Desformatação do dado: apenas repassa o dado binário.

Parâmetros sequencia – bytes de dados

Retorna o dado sem a formatação

Dado de comprimento fixo

class estrutarq.dado.DadoFixo(comprimento: int, preenchimento=b\xff')

Base: estrutarq.dado.dado_comum.DadoBasico

Classe dado de comprimento fixo

 $adicione_formatacao(dado: bytes) \rightarrow bytes$

Formatação do dado: ajusta o dado para o comprimento definido, truncando ou adicionando o byte de preenchimento.

Parâmetros dado – valor do dado

Retorna o dado formatado no comprimento especificado

leia_de_arquivo(arquivo: BinaryIO) \rightarrow bytes

Leitura de um único dado de comprimento fixo a partir do arquivo :param arquivo: arquivo binário aberto com permissão de leitura.

Retorna os bytes do campo

Os bytes de preenchimento que existirem são removidos.

leia_de_bytes(sequencia: bytes) -> (<class 'bytes'>, <class 'bytes'>)

Recuperação de um dado individual de uma sequência de bytes, retornando o dado sem os bytes de preenchimento e o restante da sequência.

Parâmetros sequencia – uma sequência de bytes

Retorna tupla com os bytes do dado, removidos os bytes de organização de dados, e a sequência de bytes restante

property preenchimento: bytes

remova_formatacao(sequencia: bytes) \rightarrow bytes

Desformatação do dado: remoção de caracteres de preenchimento.

Parâmetros sequencia – bytes de dados

Retorna dado efetivo, sem preenchimento

Dado básico

Todas as demais classes do módulo são derivadas de uma classe abstrata básica.

class estrutarq.dado.DadoBasico

Base: object

Classe básica para armazenamento e manipulação de dados.

Implementa as operações básicas e define os métodos abstratos.

abstract adicione_formatacao(dado: bytes) \rightarrow bytes

Acréscimo da organização de dados em uso aos bytes do dado.

Parâmetros dado (*bytes*) – bytes do dado

Retorna bytes do dado acrescido da forma de organização

Tipo de retorno bytes

byte_enchimento: bytes = $b'\x1b'$

Contém o byte de escape usado para enchimento (byte stuffing). Valor padrão: ESC (hexadecimal 0x1B).

enchimento_de_bytes(sequencia: bytes, lista_bytes: list[bytes]) → bytes

Operação de enchimento de bytes (*byte stuffing*). Antes de cada item de lista_bytes é acrescentado o byte byte_enchimento.

Parâmetros

- **sequencia** (*bytes*) a sequência de bytes a ser "enchida"
- lista_bytes (list[bytes]) os bytes especiais que serão "escapados"

Retorna a sequência original enchida

Tipo de retorno bytes

esvaziamento_de_bytes(sequencia: bytes) \rightarrow bytes

Operação de esvaziamento de bytes (*byte un-stuffing*). Todos os enchimentos feitos com byte_enchimento são removidos.

Parâmetros sequencia (*bytes*) – a sequência de bytes a ser "esvaziada"

Retorna a sequência sem os enchimentos

Tipo de retorno bytes

abstract leia_de_arquivo(arquivo: BinaryIO) \rightarrow bytes

Recuperação de um dado lido de um arquivo, observando a representação do dado e a forma de organização. A forma de organização usada é removida.

Parâmetros arquivo (BinaryIO) – arquivo binário aberto com permissão de leitura

Retorna a sequência de bytes lida

Tipo de retorno bytes

abstract leia_de_bytes(*sequencia: bytes*) → tuple[bytes, bytes]

Recuperação de um dado a partir de uma sequência de bytes, retornando os bytes do dado em si e o restante da sequência depois da extração do dado, observando a representação do dado e a forma de organização. O dado é retornado sem a organização.

Parâmetros sequencia (bytes) – sequência de bytes

Retorna tupla com os bytes do dado, removidos os bytes de organização de dados, e a sequência de bytes restante

Tipo de retorno tuple[bytes, bytes]

abstract remova_formatacao(sequencia: bytes) \rightarrow bytes

Remoção dos bytes correspondentes à forma de organização da sequência de bytes.

Parâmetros sequencia (*bytes*) – uma sequência de bytes

Retorna a sequência após extraídos os bytes de organização

Tipo de retorno bytes

varredura_com_enchimento($sequencia: bytes, referencia: bytes) <math>\rightarrow$ tuple[bytes, bytes]

Recuperação de um dado individual de uma sequência de bytes, retornando o dado até um byte de referência (não "enchido") e o restante da sequência depois desse byte.

Parâmetros

• **sequencia** (*bytes*) – uma sequência de bytes

• referencia (bytes) – byte simples usado como sentinela (terminador)

Retorna uma tupla contendo a sequência de bytes até referencia e o restante da sequência depois de referencia

Tipo de retorno tuple[bytes, bytes]

Levanta ValueError – se o byte de referência não estiver presente na sequência de bytes

1.1.2 estrutarq.campo package

Submodules

estrutarq.campo.campo_cadeia module

Campos para armazenamento de cadeias de caracteres.

Este arquivo provê classes para uso de campos cujo conteúdo é uma cadeia de caracteres. Internamente, o tipo *str* é usado para armazenamento e a transformação para sequência de bytes usa a codificação UTF-8.

Uma classe básica *CampoCadeiaBasico* define uma classe abstrata (ABC) com as propriedades e métodos gerais. Dela são derivadas campos:

- · Com terminadores
- Prefixada pelo comprimento
- De comprimento fixo predefinido

Licença: GNU GENERAL PUBLIC LICENSE V.3, 2007

Jander Moreira, 2021-2022

```
class estrutarq.campo.campo_cadeia.CampoCadeiaBasico(tipo: str, valor: str = ")
```

Base: estrutarq.campo.campo_comum.CampoBasico

Classe básica para cadeias de caracteres.

Parâmetros

- **tipo** (*str*) nome do tipo (definido nas classes derivadas)
- valor (str, opcional) o valor a ser armazenado no campo (padrão: "")

bytes_para_valor(dado: bytes)

Armazenamento da sequência de bytes de dado como valor do campo.

Parâmetros dado (bytes) – sequência de bytes com codificação UTF-8

property valor

Recuperação, com as devidas conversões, do atributo __valor :return: o valor de __valor

```
valor_para_bytes() \rightarrow bytes
```

Retorno do valor do campo convertido para sequência de bytes usando codificação UTF-8.

Retorna sequência de bytes

Tipo de retorno bytes

```
class estrutarq.campo_cadeia.CampoCadeiaFixo(comprimento: int, **kwargs)
     Base:
                      estrutarq.dado.dado_comum.DadoFixo,
                                                                     estrutarq.campo.campo_cadeia.
     CampoCadeiaBasico
     Classe para cadeia de caracteres com comprimento bloco fixo e preenchimento de dados inválidos
class estrutarq.campo.campo_cadeia.CampoCadeiaPrefixado(*args, **kwargs)
     Base:
                  estrutarq.dado.dado_comum.DadoPrefixado,
                                                                     estrutarq.campo.campo_cadeia.
     CampoCadeiaBasico
     Classe para cadeia de caracteres prefixada pelo comprimento_bloco
class estrutarq.campo.campo_cadeia.CampoCadeiaTerminador(**kwargs)
     Base:
                  estrutarq.dado.dado_comum.DadoTerminador,
                                                                     estrutarq.campo.campo_cadeia.
     CampoCadeiaBasico
     Classe para cadeia de caracteres com terminador
          Parâmetros kwargs (:class:dict) – parâmetros nomeados a serem repassados
estrutarq.campo.campo comum module
class estrutarq.campo.campo_comum.CampoBasico(tipo: str)
     Base: estrutarq.dado.dado_comum.DadoBasico
     Estruturação básica do campo como menor unidade de informação.
          Parâmetros tipo (str) – cadeia de caracteres com o nome do tipo
     abstract bytes_para_valor(dado: bytes)
          Conversão de uma sequência de bytes para armazenamento no valor do campo, de acordo com a represen-
          tação de dados :param dado: sequência de bytes :return: o valor do campo de acordo com seu tipo
     comprimento()
          Obtém o comprimento atual do campo
              Retorna o comprimento do campo
     comprimento_fixo()
          Retorna se o comprimento é ou não fixo
              Retorna True para comprimento fixo ou False para variável
     copy()
          Cópia "rasa" deste campo :return: outra instância com os mesmos valores
     escreva(arquivo: BinaryIO)
          Conversão do valor para sequência de bytes e armazenamento no arquivo
              Parâmetros arquivo – arquivo binário aberto com permissão de escrita
     leia(arquivo: BinaryIO)
          Conversão dos dado lidos para o valor do campo, obedecendo à organização e formato de representação
              Parâmetros arquivo – arquivo binário aberto com permissão de leitura
     property tipo
     abstract property valor
          Recuperação, com as devidas conversões, do atributo __valor :return: o valor de __valor
```

```
abstract valor_para_bytes() → bytes
```

Conversão do valor do campo para sequência de bytes de acordo com a representação de dados :return:

```
class estrutarq.campo.campo_comum.CampoBruto(valor=")
```

```
Base: estrutarq.dado.dado_comum.DadoBruto, estrutarq.campo.campo_comum.CampoBasico
```

Implementação das funções de um campo bruto, ou seja, sem organização de campo. O valor é sempre armazenado como cadeia de caracteres.

```
bytes_para_valor(dado: bytes)
```

Conversão de sequência de bytes para valor o campo, considerando uma cadeia de caracteres simples :param dado: a sequência de bytes

property valor

Recuperação, com as devidas conversões, do atributo __valor :return: o valor de __valor

```
valor_para_bytes() \rightarrow bytes
```

Conversão do valor do campo (cadeia de caracteres) para uma sequência de bytes, usando codificação UTF-8 :return: a sequência de bytes

estrutarq.campo.campo inteiro module

```
class estrutarq.campo_inteiro.CampoIntBasico(tipo: str, valor: int = 0)
```

Base: estrutarq.campo.campo_comum.CampoBasico

Classe básica para campo inteiro

```
bytes_para_valor(dado: bytes)
```

Conversão de uma sequência de bytes (representação textual) para inteiro :param dado: sequência de bytes

```
property valor: int
```

Recuperação, com as devidas conversões, do atributo __valor :return: o valor de __valor

```
valor_para_bytes() \rightarrow bytes
```

Conversão do valor inteiro para sequência de bytes usando representação textual e codificação UTF-8 :return: sequência de bytes

```
class estrutarq.campo.campo_inteiro.CampoIntBinario(**kwargs)
```

```
Base: estrutarq.dado.dado_comum.DadoBinario, estrutarq.campo.campo_inteiro.
CampoIntBasico
```

Classe para inteiro em formato binário (big endian) com 8 bytes e complemento para 2 para valores negativos

```
bytes_para_valor(dado: bytes)
```

Conversão de uma sequência de bytes (binária big-endian com sinal) para inteiro :param dado: sequência de bytes

```
numero_bytes = 8
```

```
valor_para_bytes() \rightarrow bytes
```

Conversão do valor inteiro para sequência de bytes usando representação binária big-endian com sinal :return: sequência de bytes

```
class estrutarq.campo.campo_inteiro.CampoIntFixo(comprimento: int, **kwargs)
```

```
Base: estrutarq.dado.dado_comum.DadoFixo, estrutarq.campo.campo_inteiro.CampoIntBasico
```

Classe para inteiro textual com tamanho fixo

```
class estrutarq.campo.campo_inteiro.CampoIntPrefixado(**kwargs)
                 estrutarq.dado.dado_comum.DadoPrefixado,
                                                                    estrutarq.campo.campo_inteiro.
     CampoIntBasico
     Classe para inteiro textual com prefixo de comprimento bloco
class estrutarq.campo.campo_inteiro.CampoIntTerminador(terminador: bytes = b'\x00', **kwargs)
     Base:
                 estrutarg.dado.dado_comum.DadoTerminador,
                                                                   estrutarg.campo.campo_inteiro.
     CampoIntBasico
     Classe para inteiro textual com terminador
estrutarq.campo.campo real module
class estrutarq.campo.campo_real.CampoRealBasico(tipo: str, valor: float = 0)
     Base: estrutarq.campo.campo_comum.CampoBasico
     Classe básica para campo real
     bytes_para_valor(dado: bytes)
          Conversão de sequência de bytes com valor textual para valor real :param dado: sequência de 8 bytes
     property valor: float
          Recuperação, com as devidas conversões, do atributo __valor :return: o valor de __valor
     valor_para_bytes() \rightarrow bytes
          Conversão do valor do campo para sequência de bytes textual :return: a sequência de bytes no padrão
          especificado
class estrutarq.campo.campo_real.CampoRealBinario(**kwargs)
                     estrutarq.dado.dado_comum.DadoBinario,
                                                                       estrutarq.campo.campo_real.
     CampoRealBasico
     Classe para real em formato binário usando IEEE 754 de precisão dupla
     bytes_para_valor(dado: bytes)
          Conversão de sequência de bytes com representação IEEE 754 de precisão dupla para real :param dado:
          sequência de 8 bytes
     valor_para_bytes() \rightarrow bytes
          Conversão do valor do campo para sequência de bytes no padrão IEEE 754 de precisão dupla :return: a
          sequência de bytes no padrão especificado
class estrutarq.campo.campo_real.CampoRealFixo(comprimento: int, **kwargs)
     Base: estrutarq.dado.dado_comum.DadoFixo, estrutarq.campo.campo_real.CampoRealBasico
     Classe para campo real com representação textual de comprimento_bloco fixo
class estrutarq.campo.campo_real.CampoRealPrefixado(**kwargs)
                   estrutarq.dado.dado_comum.DadoPrefixado,
                                                                       estrutarq.campo.campo_real.
     CampoRealBasico
     Classe para campo real com representação textual de comprimento bloco fixo
class estrutarq.campo.campo_real.CampoRealTerminador(terminador: bytes = b \times 00', **kwargs)
                   estrutarg.dado.dado_comum.DadoTerminador,
                                                                       estrutarg.campo.campo_real.
     CampoRealBasico
```

Classe para campo real com representação textual de comprimento_bloco fixo

estrutarq.campo.campo tempo module

```
class estrutarq.campo.campo_tempo.CampoDataBinario(**kwargs)
```

Base: estrutarq.dado.dado_comum.DadoBinario, estrutarq.campo.campo_tempo. CampoTempoBasicoBinario

Classe para armazenamento de data (dia, mês e ano) para armazenamento em formato binário.

```
class estrutarq.campo.campo_tempo.CampoDataFixo(**kwargs)
```

```
Base: estrutarq.dado.dado_comum.DadoFixo, estrutarq.campo.campo_tempo.
CampoTempoBasicoFixo
```

Classe para data, em número de segundos desde 1/1/1970, 0h00min00s usando armazenamento em cadeia de caracteres no formato 'formato_data'.

```
class estrutarq.campo.campo_tempo.CampoHoraBinario(**kwargs)
```

```
Base: estrutarq.dado.dado_comum.DadoBinario, estrutarq.campo.campo_tempo.
CampoTempoBasicoBinario
```

Classe para horário usando armazenamento em valor inteiro em binário, com sinal, big-endian.

```
class estrutarq.campo.campo_tempo.CampoHoraFixo(**kwargs)
```

```
Base: estrutarq.dado.dado_comum.DadoFixo, estrutarq.campo.campo_tempo.
CampoTempoBasicoFixo
```

Classe horário usando armazenamento em cadeia de caracteres no formato 'formato hora'.

```
class estrutarq.campo.campo_tempo.CampoTempoBasico(tipo: str, formato: str, apenas\_data: bool, valor: <math>str = ", **kwargs)
```

```
Base: estrutarq.campo.campo_comum.CampoBasico
```

Base: estrutarq.campo.campo_tempo.CampoTempoBasico
Implementação das conversões tempo-> binário e binário->tempo

Classe básica para campo de tempo (data + horário), armazenado internamente como o número de segundos desde 1/1/1970, 0h00min00s.

Quando apenas a data é armazenada, o horário é ajustado para 12h00min00s, para evitar problemas com fuso horário

```
comprimento_data = 10

comprimento_hora = 8

comprimento_tempo = 19

formato_data = '%Y-%m-%d'

formato_hora = '%H:%M:%S'

formato_tempo = '%Y-%m-%d %H:%M:%S'

property segundos: int

property valor: str

Recuperação, com as devidas conversões, do atributo __valor :return: o valor de __valor class estrutarq.campo.campo_tempo.CampoTempoBasicoBinario(*args, **kwargs)
```

1.1. Conteúdo geral

```
bytes_para_valor(dado: bytes)
```

Conversão da representação binária (8 bytes, big-endian, com sinal) para valor inteiro de segundos :param dado: bytes da representação do inteiro em binário

```
valor_para_bytes() \rightarrow bytes
```

Conversão do valor do tempo em segundos para representação em inteiro binário (8 bytes, big-endian, com sinal) :return: a sequência de bytes

```
class estrutarq.campo.campo_tempo.CampoTempoBasicoFixo(*args, **kwargs)
```

```
Base: estrutarq.campo.campo_tempo.CampoTempoBasico
```

Implementação das conversões tempo-> binário e binário->tempo

```
bytes_para_valor(dado: bytes)
```

Conversão da representação binária (8 bytes, big-endian, com sinal) para valor inteiro de segundos :param dado: bytes da representação do inteiro em binário

```
valor_para_bytes() \rightarrow bytes
```

Conversão do valor do tempo em segundos para representação em inteiro binário (8 bytes, big-endian, com sinal) :return: a sequência de bytes

```
class estrutarq.campo.campo_tempo.CampoTempoBinario(**kwargs)
```

```
Base: estrutarq.dado.dado_comum.DadoBinario, estrutarq.campo.campo_tempo.
CampoTempoBasicoBinario
```

Classe para tempo (data + horário), em número de segundos desde 1/1/1970, 0h00min00s usando armazenamento em valor inteiro em binário, com sinal, big-endian.

```
class estrutarq.campo.campo_tempo.CampoTempoFixo(**kwargs)
```

```
Base: estrutarq.dado.dado_comum.DadoFixo, estrutarq.campo.campo_tempo.
CampoTempoBasicoFixo
```

Classe para tempo (data + horário), em número de segundos desde 1/1/1970, 0h00min00s usando armazenamento em cadeia de caracteres no formato 'formato_tempo'.

Module contents

Módulo: campo

Implementação de representações de campos.

1.1.3 Pacote estrutarq.registro

Módulo estrutarq.registro.registro_comum

Registros

```
class estrutarq.registro_registro_comum.RegistroBasico(tipo: str, *lista_campos)
```

```
Base: estrutarq.dado.dado_comum.DadoBasico
```

Classe básica para registros

Utiliza @DynamicAttrs

adicione_campos(*lista_campos)

Inclusão de uma sequência de campos ao registro :param lista_campos: uma sequência de um ou mais campos, cada um

especificado pela tupla (nome_arquivo, campo), com nome_arquivo (str) sendo o nome_arquivo do campo e campo sendo uma instância de um campo válido

comprimento()

Retorna o comprimento do registro em bytes caso ele tenha comprimento total fixo :return: o comprimento do registro em bytes ou None se tiver comprimento variável

copy()

Cópia "profunda" deste campo :return: outra instância com os mesmos valores

de_bytes(dados_registro: bytes)

Obtenção dos bytes de cada campo a partir dos bytes do registro inteiro :param dados_registro: sequência de bytes do registro

escreva(arquivo)

Escrita do registro no arquivo :param arquivo:

leia(arquivo)

Obtenção de um registro a partir do arquivo :param arquivo: arquivo binário aberto com permissão de leitura

$para_bytes() \rightarrow bytes$

Criação dos bytes do registro pela concatenação dos bytes dos campos, sucessivamente :return: sequência dos bytes dos campos

tem_comprimento_fixo()

Verifica se o registro tem comprimento fixo :return: True se o comprimento for fixo

O registro é considerado de tamanho fixo se qualquer uma das propriedades foram verdadeiras:

- 1) o registro tem é marcado com _comprimento_fixo == True
- 2) todos os campos tiverem comprimento fixo

property tipo

class estrutarq.registro.registro_comum.RegistroBruto(*lista_campos)

```
Base: estrutarq.dado.dado\_comum.DadoBruto, estrutarq.registro\_registro\_comum. \\ RegistroBasico
```

Classe básica para registro, com controle exclusivamente pelo número de campos

Utiliza @DynamicAttrs

class estrutarq.registro_registro_comum.RegistroFixo(comprimento: int, *lista_campos)

```
 \begin{array}{lll} \textbf{Base:} & \textit{estrutarq.dado.dado\_comum.DadoFixo}, & \textit{estrutarq.registro.registro\_comum.} \\ \textit{RegistroBasico} \end{array}
```

Classe para registros com terminador

class estrutarq.registro.registro_comum.RegistroPrefixado(*lista_campos)

```
Base: estrutarq.dado.dado\_comum.DadoPrefixado, estrutarq.registro.registro\_comum. \\ RegistroBasico
```

Classe para registros prefixados pelo comprimento

class estrutarq.registro.registro_comum.RegistroTerminador(*lista_campos)

 $Base: estrutarq.dado.dado_comum.DadoTerminador, estrutarq.registro.registro_comum. \\ RegistroBasico$

Classe para registros com terminador

1.1.4 estrutarq.arquivo package

Submodules

estrutarq.arquivo.arquivo_comum module

Base: object

Gerenciador dedicado a um único arquivo aberto

abstract escreva(registro: estrutarq.registro.registro_comum.RegistroBasico)

Gravação de um registro no arquivo

feche()

Fechamento do arquivo associado

abstract leia() → *estrutarq.registro.registro_comum.RegistroBasico*

Leitura de um registro do arquivo :return: o registro lido

posicao_atual()

Posição atual do arquivo :return:

Base: estrutarq.arquivo.arquivo_comum.ArquivoBasico

Gerenciador de arquivo simples (como fluxo de dados) com registros de comprimento fixo.

escreva(registro: estrutarq.registro.registro_comum.RegistroBasico, **kwargs)

Gravação de um registro no arquivo

self.escreva_efetivo chama escreva_fixo ou escreva_variável, conforme o registro tenha comprimento fixo ou variável

escreva_fixo(*registro*: estrutarq.registro_comum.RegistroBasico, *posicao_relativa*: Optional[int] = None)

Gravação de um registro no arquivo :param registro: o registro a ser escrito :param posicao_relativa: posição relativa do registro no arquivo,

com o primeiro registro sendo o registro 0

escreva_variavel(*registro*: estrutarq.registro_comum.RegistroBasico, *deslocamento*: *Optional[int]* = *None*)

Gravação de um registro no arquivo :param registro: o registro a ser escrito :param deslocamento: posição absoluta (byte offset) da posição

de escrita

leia(**kwargs) → estrutarq.registro.registro_comum.RegistroBasico

Leitura de um registro do arquivo :return: o registro lido

self.leia_efetivo chama leia_fixo ou leia_variável, conforme o registro tenha comprimento fixo ou variável

 $leia_fixo(posicao_relativa: Optional[int] = None) \rightarrow estrutarq.registro.registro_comum.RegistroBasico$

Leitura de um registro de comprimento fixo :param posicao_relativa: posição relativa do registro no arquivo,

com o primeiro registro sendo o registro 0

Retorna o registro lido

```
leia_variavel(posicao_relativa: Optional[int] = None) →
estrutarq.registro.registro_comum.RegistroBasico
```

Leitura de um registro de comprimento variavel :param posicao_relativa: posição relativa do registro no arquivo,

com o primeiro registro sendo o registro 0

Retorna o registro lido

A determinação da posição relativa é feita por busca sequencial

Module contents

1.1.5 estrutarq.utilitarios package

Submodules

estrutarq.utilitarios.disco module

Rotinas utilitárias gerais

```
estrutarq.utilitarios.disco.comprimento_de_bloco(diretorio: Optional[str] = None)
```

Determina o comprimento_bloco de um bloco de disco, tendo como referência o disco onde está o diretório temporário do sistema; para outro disco, é preciso informar um diretório nesse disco em que haja direito de criação de arquivos. :param diretorio: um diretório no disco a ser verificado :return: o tamanho do bloco no disco

Efeitos colaterais: é criado um arquivo temporário, que em seguida é removido.

estrutarq.utilitarios.disco.main()

estrutarq.utilitarios.dispositivo module

Rotinas utilitárias gerais

```
estrutarq.utilitarios.dispositivo.comprimento_de_bloco(diretorio: Optional[str] = None)
```

Determina o comprimento_bloco de um bloco do dispositivo externo, tendo como referência aquele onde está o diretório temporário do sistema (parametro igual a None); se um diretório em que haja direito de criação de arquivos for informado, então o dispostivo em que ele está será utilizado. :param diretorio: um diretório no disco a ser verificado :return: o tamanho do bloco no disco

Efeitos colaterais: é criado um arquivo temporário, que em seguida é removido.

Estruturas de arquivos: estrutarq, Release 0.1

estrutarq.utilitarios.dispositivo.main()

estrutarq.utilitarios.geral module

Funções gerais

estrutarq.utilitarios.geral.verifique_versao()

estrutarq.utilitarios.interpretador module

Module contents

Índice de Módulos Python

е

```
estrutarq.arquivo, 15
estrutarq.arquivo.arquivo_comum, 14
estrutarq.campo, 12
estrutarq.campo.campo_cadeia, 7
estrutarq.campo.campo_comum, 8
estrutarq.campo.campo_inteiro, 9
estrutarq.campo.campo_real, 10
estrutarq.campo.campo_tempo, 11
estrutarq.dado.dado_comum, 1
estrutarq.registro.registro_comum, 12
estrutarq.utilitarios, 16
estrutarq.utilitarios.disco, 15
estrutarq.utilitarios.dispositivo, 15
estrutarq.utilitarios.geral, 16
```

A	bytes_para_valor() (método estru-
adicione_campos() (método estru-	tarq.campo.campo_real.CampoRealBasico),
tarq.registro.registro_comum.RegistroBasico),	10 bytes_para_valor() (método estru-
adicione_formatacao() (método estru- tarq.dado.DadoBasico), 5	tarq.campo.campo_real.CampoRealBinario),
adicione_formatacao() (método estru- tarq.dado.DadoBinario), 4	$bytes_para_valor() \qquad (\textit{m\'etodo} \qquad \textit{estru-} \\ \textit{tarq.campo.campo_tempo.CampoTempoBasicoBinario}),$
adicione_formatacao() (método estru- tarq.dado.DadoBruto), 2	bytes_para_valor() (método estru-
adicione_formatacao() (método estru- tarq.dado.DadoFixo), 5	tarq.campo.campo_tempo.CampoTempoBasicoFixo), 12
adicione_formatacao() (método estru- tarq.dado.DadoPrefixado), 3	C
adicione_formatacao() (método estru-	CampoBasico (classe em estru- tarq.campo.campo_comum), 8
tarq.dado.DadoTerminador), 2 ArquivoBasico (classe em estru-	CampoBruto (classe em estru-
tarq.arquivo.arquivo_comum), 14	tarq.campo.campo_comum), 9
ArquivoSimples (classe em estru- tarq.arquivo.arquivo_comum), 14	CampoCadeiaBasico (classe em estru- tarq.campo.campo_cadeia), 7
_	CampoCadeiaFixo (classe em estru-
В	tarq.campo.campo_cadeia), 7 CampoCadeiaPrefixado (classe em estru-
byte_enchimento (atributo estru- tarq.dado.DadoBasico), 5	tarq.campo.campo_cadeia), 8
bytes_para_valor() (método estru- tarq.campo.campo_cadeia.CampoCadeiaBasico)	CampoCadeiaTerminador (classe em estru- tarq.campo.campo_cadeia), 8
7 bytes_para_valor() (método estru-	''CampoDataBinario (classe em estru- tarq.campo.campo_tempo), 11
tarq.campo.campo_comum.CampoBasico),	CampoDataFixo (classe em estru- tarq.campo.campo_tempo), 11
bytes_para_valor() (método estru-	CampoHoraBinario (classe em estru- tarq.campo.campo_tempo), 11
tarq.campo.campo_comum.CampoBruto),	CampoHoraFixo (classe em estru- tarq.campo.campo_tempo), 11
bytes_para_valor() (método estru- tarq.campo.campo_inteiro.CampoIntBasico),	CampoIntBasico (classe em estru- tarq.campo.campo_inteiro), 9
bytes_para_valor() (método estru- tarq.campo.campo_inteiro.CampoIntBinario),	CampoIntBinario (classe em estru- tarq.campo.campo_inteiro), 9
9	CampoIntFixo (classe em estru- tarq.campo.campo_inteiro), 9

CampoIntPrefixado (classe em estru-	D
tarq.campo.campo_inteiro),9 CampoIntTerminador (classe em estru-	DadoBasico (classe em estrutarq.dado), 5 DadoBinario (classe em estrutarq.dado), 4
tarq.campo.campo_inteiro), 10 CampoRealBasico (classe em estru-	DadoBruto (classe em estrutarq.dado), 2
tarq.campo.campo_real), 10	DadoPrefixado (classe em estrutarq.dado), 3
CampoRealBinario (classe em estru- tarq.campo.campo_real), 10	DadoTerminador (classe em estrutarq.dado), 2 de_bytes() (método estru-
CampoRealFixo (classe em estru- tarq.campo.campo_real), 10	$tarq.registro.registro_comum.RegistroBasico),$
CampoRealPrefixado (classe em estru-	13 E
tarq.campo.campo_real), 10 CampoRealTerminador (classe em estru-	-
tarq.campo_real), 10	tarq.dado.DadoBasico), 6
CampoTempoBasico (classe em estru-	
tarq.campo.campo_tempo), 11	$tarq. arquivo. arquivo_comum. ArquivoBasico),$
CampoTempoBasicoBinario (classe em estru-	14
tarq.campo.campo_tempo), 11	escreva() (método estru-
CampoTempoBasicoFixo (classe em estru- tarq.campo.campo_tempo), 12	tarq.arquivo.arquivo_comum.ArquivoSimples), 14
CampoTempoBinario (classe em estru-	escreva() (método estru-
tarq.campo.campo_tempo), 12	tarq.campo.campo_comum.CampoBasico),
CampoTempoFixo (classe em estru-	8
tarq.campo.campo_tempo), 12	escreva() (método estru-
comprimento() (método estru-	$tarq.registro.registro_comum.RegistroBasico),$
tarq.campo.campo_comum.CampoBasico), 8	13
comprimento() (método estru-	escreva_fixo() (método estru-
tarq.registro.registro_comum.RegistroBasico),	tarq.arquivo.arquivo_comum.ArquivoSimples),
13	escreva_variavel() (método estru-
comprimento_data (atributo estru-	
tarq.campo.campo_tempo.CampoTempoBasico), 14
comprimento_de_bloco() (no módulo estru-	estrutarq.arquivo módulo,15
tarq.utilitarios.disco), 15 comprimento_de_bloco() (no módulo estru-	estrutarq.arquivo.arquivo_comum
tarq.utilitarios.dispositivo), 15	modulo, 14
comprimento_fixo() (método estru-	estrutarq.campo módulo,12
tarq.campo.campo_comum.CampoBasico), 8	estrutarq.campo.campo_cadeia
comprimento_hora (atributo estru-	módulo, 7
tarq.campo.campo_tempo.CampoTempoBasico	estrutaro, campo campo comum
11 comprimento_tempo (<i>atributo estru</i> -	estrutarq.campo.campo_inteiro
tarq.campo.campo_tempo.CampoTempoBasico	modulo, 9
ин q.ситро.ситро_тетро.ситро1етроВизисо 11	estrutary.campo.campo_rear
copy() (método estru-	módulo, 10
tarq.campo.campo_comum.CampoBasico),	estrutarq.campo.campo_tempo módulo,11
Sony() (mátodo actru	estrutarq.dado.dado_comum
copy() (método estru-	modulo, 1
tarq.registro.registro_comum.RegistroBasico), 13	estrutarq.registro.registro_comum módulo,12
	estrutarq.utilitarios
	módulo, 16

estruta	rq.utilitarios.di	SC0		<pre>leia_de_bytes() (método</pre>	o estrutarq.dado.Dad	doFixo), 5
	ulo, 15			<pre>leia_de_bytes()</pre>	(método	estru-
	rq.utilitarios.di	spositivo		tarq.dado.DadoP.	refixado), 3	
	ulo, 15			<pre>leia_de_bytes()</pre>	(método	estru-
	rq.utilitarios.ge	ral		tarq.dado.DadoTe	erminador), 3	
	ulo, 16			<pre>leia_fixo()</pre>	(método	estru-
esvazia	mento_de_bytes() tarq.dado.DadoBasic	(método o), 6	estru-	tarq.arquivo.arqu 15	ivo_comum.Arquivo	Simples),
_				<pre>leia_variavel()</pre>	(método	estru-
F				tarq.arquivo.arqu	ivo_comum.Arquivo	Simples),
feche()	(méto	odo	estru-	15		
	tarq.arquivo.arquivo_ 14	_comum.ArquivoBa	ısico),	M		
formato	o_data (ar	tributo	estru-	main() (no módulo estruta	rq.utilitarios.disco),	15
	tarq.campo.campo_te			main() (no módulo estruta módulo	rq.utilitarios.disposi	itivo), 15
formato		tributo	estru-	estrutarq.arquivo,	, 15	
	tarq.campo.campo_te			estrutarq.arquivo		14
	11	T · · · · · T · · · · T ·	,,,	estrutarq.campo, 12		
formato	_tempo (a	atributo	estru-	estrutarq.campo.ca	ampo_cadeia,7	
	tarq.campo.campo_te	тро.СатроТетро	Basico),	estrutarq.campo.ca	$ampo_comum, 8$	
	11			estrutarq.campo.ca	${\tt ampo_inteiro}, 9$	
				estrutarq.campo.ca	ampo_real,10	
L				estrutarq.campo.ca		
leia()	(métod	do	estru-	estrutarq.dado.da		
	tarq.arquivo.arquivo_	_comum.ArquivoBa	ısico),	estrutarq.registro.registro_comum, 12		
	14			estrutarq.utilita		
leia()	(métod	do	estru-	estrutarq.utilita		
	tarq.arquivo.arquivo_	comum.ArquivoSir	nples),	estrutarq.utilita		5, 15
	14	•		estrutarq.utilita	rios.gerai, io	
leia()	(métod		estru-	N		
	tarq.campo.campo_co	этит.СатроВаѕіс	co),		(, '1 ,	
1-:-()	8	1.	4	numero_bytes	(atributo	estru-
leia()	(métod		estru-	iarq.campo.campo	o_inteiro.CampoIntl	sinario),
	tarq.registro.registro_	_comum.kegistroBe	asico),	9		
leia_de	_arquivo()	(método	estru-	Р		
	tarq.dado.DadoBasic			para_bytes()	(método	estru-
leia_de	_arquivo()	(método	estru-		stro_comum.Registro	oBasico),
	tarq.dado.DadoBinar	<i>io</i>), 4		13		
leia_de	_arquivo()	(método	estru-	<pre>posicao_atual()</pre>	(método	estru-
	tarq.dado.DadoBruto), 2		tarq.arquivo.arqu	ivo_comum.Arquivo	Basico),
leia_de	_arquivo()	(método	estru-	14		
	tarq.dado.DadoFixo),			preenchimento (propried	ade estrutarq.dado	DadoFixo
leia_de	_arquivo()	(método	estru-), 5		
	tarq.dado.DadoPrefix			В		
leia_de	_arquivo()	(método	estru-	K		
,	tarq.dado.DadoTermi	, ·		•	classe em	estru-
rela_de	_ , ,,	(método	estru-	tarq.registro.regis		
10:5-1	tarq.dado.DadoBasic		ac+	•	lasse em	estru-
rera_de	_ , ,,	(método	estru-	tarq.registro.regis		
lois da	tarq.dado.DadoBinar _bytes() (método es		Rruto	•	asse em	estru-
rera_ue	:_by ces() (meiouo es	п шигү.иию.⊅ии0	, (מוזותי	tarq.registro.regis	stro_comum), 13	
	_					

Registr	coPrefixado (classe em tarq.registro.registro_comum), 13	estru-	valor_para_bytes() (método estru- tarq.campo.campo_cadeia.CampoCadeiaBasico),
Registi	coTerminador (classe em tarq.registro.registro_comum), 13	estru-	7 valor_para_bytes() (método estru-
remova_	_formatacao() (método tarq.dado.DadoBasico), 6	estru-	tarq.campo.campo_comum.CampoBasico), 8
remova_	_formatacao() (método tarq.dado.DadoBinario), 4	estru-	valor_para_bytes() (método estru- tarq.campo.campo_comum.CampoBruto),
remova_	_formatacao() (<i>método</i>	estru-	9
remova_	tarq.dado.DadoBruto), 2 _formatacao() (método _tara dado.DadoFiro) 5	estru-	valor_para_bytes() (método estru- tarq.campo.campo_inteiro.CampoIntBasico),
remova_	tarq.dado.DadoFixo), 5 _formatacao() (método	estru-	valor_para_bytes() (método estru-
remova_	tarq.dado.DadoPrefixado), 4 _formatacao() (método	estru-	tarq.campo.campo_inteiro.CampoIntBinario), 9
	tarq.dado.DadoTerminador), 3		valor_para_bytes() (método estru-
S			tarq.campo.campo_real.CampoRealBasico), 10
segundo	os (propriedade tarq.campo.campo_tempo.CampoTempol), 11	estru- Basico	<pre>valor_para_bytes() (método estru- tarq.campo.campo_real.CampoRealBinario), 10</pre>
Т			valor_para_bytes() (método estru- tarq.campo.campo_tempo.CampoTempoBasicoBinario),
-	mprimento_fixo() (método	estru-	12
	tarq.registro.registro_comum.RegistroBa	sico),	valor_para_bytes() (método estru- tarq.campo.campo_tempo.CampoTempoBasicoFixo), 12
termina	ador (propriedade tarq.dado.DadoTerminador), 3	estru-	varredura_com_enchimento() (método estru-
tipo	(propriedade	estru-	tarq.dado.DadoBasico), 6 verifique_versao() (no módulo estru-
	tarq.campo.campo_comum.CampoBasico), 8)	tarq.utilitarios.geral), 16
tipo	(propriedade tarq.registro.registro_comum.RegistroBa	estru-	
	13	sico),	
V			
valor	(propriedade tarq.campo.campo_cadeia.CampoCadeia), 7	estru- aBasico	
valor	(propriedade	estru-	
	tarq.campo.campo_comum.CampoBasico), 8	9	
valor	(propriedade tarq.campo.campo_comum.CampoBruto 9	estru-),	
valor	(propriedade tarq.campo.campo_inteiro.CampoIntBas	estru- ico),	
valor	(propriedade tarq.campo.campo_real.CampoRealBasion), 10	estru- co	
valor	(propriedade tarq.campo.campo_tempo.CampoTempol	estru- Basico	
), 11		