TripScheduler: Sua Viagem, Seu Código

Este documento apresenta a gramática EBNF completa da DSL TripScheduler, uma linguagem de domínio específico projetada para descrever roteiros de viagem, atividades diárias e orçamento . Explore como o TripScheduler simplifica o planejamento de suas aventuras.

Motivação

O Ponto de Partida: Uma Necessidade Real

> **No próximo semestre, estarei em um intercâmbio em Sydney, na Austrália.**

O planejamento de uma viagem longa como essa apresenta desafios práticos:

- **Organização de Dados:** Informações espalhadas em planilhas, notas e documentos.
- **Controle Financeiro:** Dificuldade em visualizar como os custos das atividades impactam o orçamento total.
- **Falta de uma Ferramenta Ideal:** Nenhum aplicativo parecia simples e focado o suficiente para o que eu precisava.

Visão Geral da Linguagem

Declaração de Destino

Define o destino da viagem e, opcionalmente, o país. Ex: destino "Lisboa", país="Portugal".

Período da Viagem

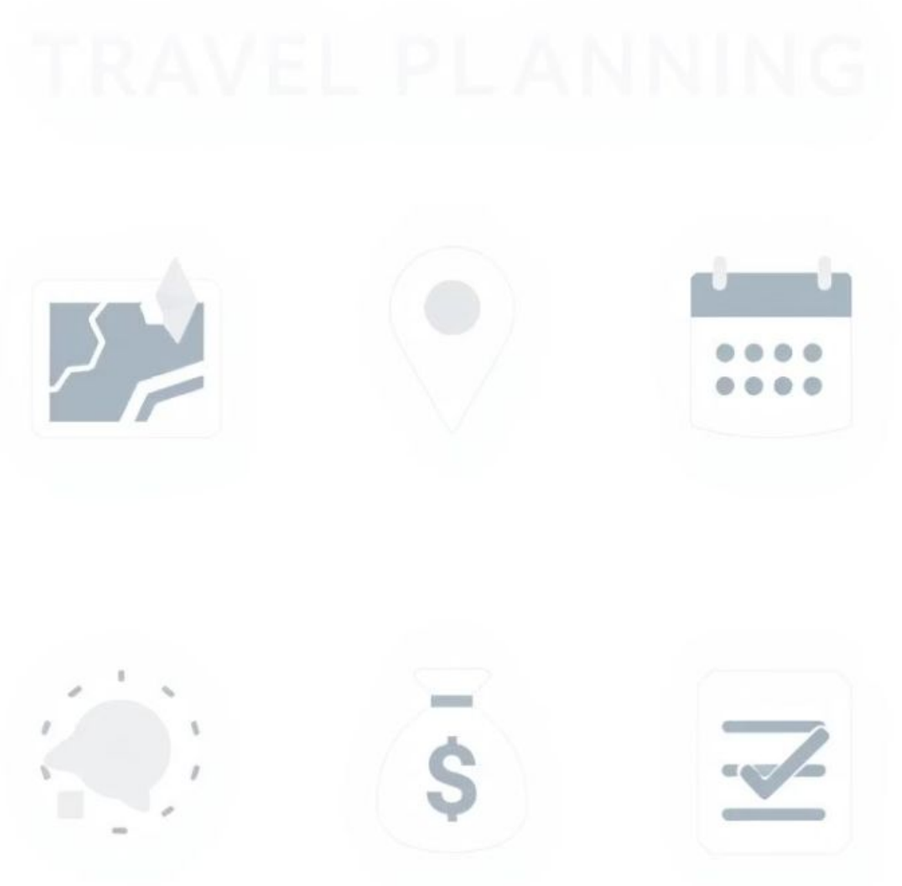
Especifica as datas de início e fim da viagem. Ex: viagem de 2025-07-10 até 2025-07-20.

Orçamento Total

Define o orçamento total em USD (apenas inteiros). Ex: budget 1500 USD.

Blocos de Atividades

Detalha atividades e custos para um dia específico ou um intervalo de dias. Ex: dia 2 { atividade "City tour" custo 30 USD }.



Gramática EBNF: A Estrutura

Programação

Um programa TripScheduler consiste em uma sequência de declarações (statement), que podem ser de destino, viagem, orçamento, bloco de dia ou

```
program = { statement } ;
```

Declarações Chave

As declarações fundamentais incluem a definição do destino, o período da viagem e o orçamento total em USD.

```
destino_decl = "destino"
string_literal ["," "país"
"=" string_literal]
;viagem_decl = "viagem" "de"
date "até" date ;budget_decl
= "budget" integer "USD" ;
```

Atividades e Custos

Blocos de dia e loops permitem detalhar atividades e seus respectivos custos, com suporte a intervalos de dias.

```
dia_block = "dia" integer "{"
{ activity_stmt cost_stmt }
"}" ;loop_stmt = "para"
"cada" "dia" "in" range "{" {
activity_stmt cost_stmt } "}"
;activity_stmt = "atividade"
string_literal ;cost_stmt =
"custo" integer "USD" ;
```

Análise Léxica com Flex

Identificação de Tokens

O Flex (trip.l) é responsável por ler o código-fonte do TripScheduler e identificar os tokens da linguagem.

Tipos de Tokens

Isso inclui palavras-chave como "destino", "viagem", "budget", literais de string ("Lisboa"), números (1500) e símbolos como chaves e ".."

Passagem para o Parser

Após a identificação, esses tokens são então passados para o analisador sintático (Bison) para a próxima fase do processo de compilação.

Análise Sintática com Bison

Definição da Gramática

O Bison (trip.y) define a gramática da linguagem, baseada na especificação EBNF, para estruturar o código.

Validação da Sintaxe

Ele valida se a sequência de tokens recebida do Flex está correta e segue as regras gramaticais estabelecidas.

Construção da Árvore

Pode ser estendido para construir uma árvore sintática abstrata (AST) ou executar ações específicas, como imprimir ou armazenar dados.

Geração e Execução

1

Geração de Arquivos

O comando **bison -d trip.y** gera **trip.tab.c** e **trip.tab.h**. Em seguida, **flex trip.l** gera **lex.yy.c**.

2

Compilação

A compilação do analisador é feita com **gcc trip.tab.c lex.yy.c -o trip_scheduler -lfl**.

3

Execução

Para executar o programa, utiliza-se **./trip_scheduler < entrada.txt**, processando o roteiro.

Arquitetura em Python

Tokenizer

Lê o código-fonte e o converte em uma sequência de tokens.

Parser

Consome os tokens e constrói a Árvore Sintática Abstrata (AST).

Processador

Atua como o cérebro da execução, coletando e armazenando dados da viagem.

Nós da AST

Classes que definem a estrutura da árvore, cada uma com um método **evaluate**.

Geração de Relatório em PDF

Relatório Detalhado

O objetivo do compilador é gerar um relatório de viagem detalhado em PDF, legível para o usuário.

Biblioteca fpdf2

O projeto utiliza a biblioteca Python **fpdf2** para a criação dos documentos PDF.

Conteúdo do PDF

O relatório inclui cabeçalho com destino e período, cronograma dia a dia de atividades e custos, e um resumo financeiro com alerta de orçamento excedido.

Como Executar o Projeto

Requisitos

Instale a dependência **fpdf2** criando um **requirements.txt** e executando **pip install -r requirements.txt**.

Criar Roteiro

Crie um arquivo de roteiro, como **entrada.txt**, com seu planejamento de viagem usando a DSL TripScheduler.

Executar Compilador

Execute o compilador via linha de comando: **python3 trip_compiler.py entrada.txt**.

Verificar Saída

Um resumo será impresso no console e um arquivo **roteiro_da_viagem.pdf** será gerado na mesma pasta.

Exemplo de Roteiro

Veja um exemplo prático de como descrever um roteiro de viagem para Tóquio usando a DSL TripScheduler.

```
// Meu roteiro de viagem dos sonhos
destino "Tóquio", país="Japão"
viagem de 2025-10-08 até 2025-10-20
budget 4000 USD
dia 8 {
     atividade "Chegada em Narita e trem para Shinjuku"
     custo 50 USD
     atividade "Jantar e explorar Shinjuku"
     custo 40 USD
} dia 9 {
     atividade "Visita ao Templo Senso-ji em Asakusa"
     custo 10 USD
     atividade "Passeio na Tokyo Skytree"
     custo 30 USD
}
para cada dia in 10..12 {
     atividade "Explorar bairros (Shibuya, Harajuku, Akihabara)"
     custo 60 USD
}
```