HELM Prompt Browser

0.1

Generated by Doxygen 1.13.2

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 HPB Namespace Reference

Enumerations

- enum class Vendor : uint8_t {
  AlephAlpha = 0x0 , ai21 = 0x1 , anthropic = 0x2 , cohere = 0x3 ,
  eleutherai = 0x4 , lmsys = 0x5 , meta = 0x6 , microsoft = 0x7 ,
  mistralai = 0x8 , mosaicml = 0x9 , openai = 0xA , stanford = 0xB ,
  tiiuae = 0xC , together = 0xD , writer_palmyra = 0xE }

Variables

- constexpr int DTColumnCount = 3
- constexpr int DTDatasetNameColumn = 0
- constexpr int DTNumberOfModels = 1
- constexpr int DTLMListColumn = 2
- constexpr int PTColumnCount = 9
- constexpr int PTCIDColumn = 0
- constexpr int PTNameIDColumn = 1
- constexpr int PTDatasetBaseColumn = 2
- constexpr int PTDatasetSpecColumn = 3
- constexpr int PTIsPromptColumn = 4
- constexpr int PTPromptContentsColumn = 5
- constexpr int PTReferencesColumn = 6
- constexpr int PTHasSpecificationsColumn = 7
- constexpr int PTIsSelectedColumn = 8
- const QList< int > list_70 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x41, 0x42, 0x50, 0x51, 0x60, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA2, 0xA3, 0xA4, 0xA5, 0xA6, 0xA7, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xD6, 0xD7, 0xD8, 0xD9, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 }
- const QList< int > list_69 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x42, 0x50, 0x51, 0x60, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA2, 0xA3, 0xA4, 0xA5, 0xA6, 0xA7, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xD6, 0xD7, 0xD8, 0xD9, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 }

- const QList< int > list_67 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x42, 0x50, 0x51, 0x60, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA4, 0xA5, 0xA6, 0xA7, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xD6, 0xD7, 0xD8, 0xD9, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 }
- const QList< int > list_66a = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x42, 0x50, 0x51, 0x60, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA4, 0xA5, 0xA6, 0xA7, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xD6, 0xD7, 0xD8, 0xD9, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0 }
- const QList< int > list_66b = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x42, 0x50, 0x51, 0x60, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA4, 0xA5, 0xA6, 0xA7, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xD6, 0xD7, 0xD8, 0xD9, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 }
- const QList< int > list_42 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x70, 0x71, 0xA0, 0xA1, 0xA4, 0xA5, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 }
- const QList< int > list_40 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x70, 0x71, 0xA0, 0xA1, 0xA4, 0xA5, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xDA, 0xDB, 0xDC, 0xDD }
- const QList< int > list_39 = { 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x70, 0x71, 0xA0, 0xA1, 0xA2, 0xA3, 0xA4, 0xA5, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xDA, 0xDB, 0xDC, 0xDD }
- const QList< int > list_32 = { 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x70, 0x71, 0xA0, 0xA1, 0xA4, 0xA5, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xD0, 0xD2, 0xD3, 0xD4, 0xD5 }
- const QList< int > list_24 = { 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x70, 0x71, 0xA0, 0xA1, 0xA4, 0xA5, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xD0, 0xD2, 0xD3, 0xD4, 0xD5 }
- const QList< int > list_10 = { 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xDA, 0xDB, 0xDC, 0xDD }
- const QList< int > list_6 = { 0x20, 0xD0, 0xD2, 0xD3, 0xD4, 0xD5 }
- const QList< int > list_2 = { 0xA2, 0xA3 }

5.1.1 Enumeration Type Documentation

5.1.1.1 Vendor

```
enum class HPB::Vendor : uint8_t [strong]
```

Enumerator

AlephAlpha	
ai21	
anthropic	
cohere	
eleutherai	
lmsys	
meta	
microsoft	
mistralai	
mosaicml	
openai	
stanford	
tiiuae	

together	
writer_palmyra	

5.1.2 Variable Documentation

5.1.2.1 DTColumnCount

```
int HPB::DTColumnCount = 3  [inline], [constexpr]
```

5.1.2.2 DTDatasetNameColumn

```
int HPB::DTDatasetNameColumn = 0  [inline], [constexpr]
```

5.1.2.3 DTLMListColumn

```
int HPB::DTLMListColumn = 2  [inline], [constexpr]
```

5.1.2.4 DTNumberOfModels

```
int HPB::DTNumberOfModels = 1  [inline], [constexpr]
```

5.1.2.5 list_10

```
const QList<int> HPB::list_10 = { 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xDA, 0xDB, 0xDC, 0xDD }
[inline]
```

5.1.2.6 list_2

```
const QList<int> HPB::list_2 = { 0xA2, 0xA3 }  [inline]
```

5.1.2.7 list_24

```
const QList<int> HPB::list_24 = { 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x70, 0x71,
0xA0, 0xA1, 0xA4, 0xA5, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xD0, 0xD2, 0xD3, 0xD4, 0xD5 }  [inline]
```

5.1.2.8 list_32

```
const QList<int> HPB::list_32 = { 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31,
0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x70, 0x71, 0xA0, 0xA1, 0xA4, 0xA5, 0xA8, 0xA9, 0xAA, 0x←
AB, 0xAC, 0xD0, 0xD2, 0xD3, 0xD4, 0xD5 }  [inline]
```

5.1.2.9 list_39

```
const QList<int> HPB::list_39 = { 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31,
0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x70, 0x71, 0xA0, 0xA1, 0xA2, 0xA3, 0xA4, 0xA5, 0xA8, 0x←
A9, 0xAA, 0xAB, 0xAC, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xDA, 0xDB, 0xDC, 0xDD }  [inline]
```

5.1.2.10 list_40

```
const QList<int> HPB::list_40 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16,
0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x70, 0x71, 0xA0, 0xA1, 0xA4, 0xA5,
0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xDA, 0xDB, 0xDC, 0xDD }
[inline]
```

5.1.2.11 list_42

```
const QList<int> HPB::list_42 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16,
0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x70, 0x71, 0xA0, 0xA1, 0xA4, 0xA5, 0x←
A8, 0xA9, 0xAA, 0xAB, 0xAC, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0,
0xE1 }  [inline]
```

5.1.2.12 list_6

```
const QList<int> HPB::list_6 = { 0x20, 0xD0, 0xD2, 0xD3, 0xD4, 0xD5 }  [inline]
```

5.1.2.13 list_66a

```
const QList<int> HPB::list_66a = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16,
0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x42, 0x50, 0x51, 0x60, 0x61,
0x62, 0x63, 0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA4, 0xA5, 0xA6, 0x←
A7, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4,
0xD5, 0xD6, 0xD7, 0xD8, 0xD9, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0 }  [inline]
```

5.1.2.14 list_66b

```
const QList<int> HPB::list_66b = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x20,
0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x42, 0x50, 0x51, 0x60, 0x61, 0x62,
0x63, 0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA4, 0xA5, 0xA6, 0xA7, 0x←
A8, 0xA9, 0xAA, 0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5,
0xD6, 0xD7, 0xD8, 0xD9, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 }  [inline]
```

5.1.2.15 list_67

```
const QList<int> HPB::list_67 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16,
0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x42, 0x50, 0x51, 0x60, 0x61,
0x62, 0x63, 0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA4, 0xA5, 0xA6, 0x←
A7, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4,
0xD5, 0xD6, 0xD7, 0xD8, 0xD9, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 }  [inline]
```

5.1.2.16 list_69

```
const QList<int> HPB::list_69 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16,
0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x42, 0x50, 0x51, 0x60, 0x61,
0x62, 0x63, 0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA2, 0xA3, 0xA4, 0x←
A5, 0xA6, 0xA7, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2,
0xD3, 0xD4, 0xD5, 0xD6, 0xD7, 0xD8, 0xD9, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 }  [inline]
```

5.1.2.17 list_70

```
const QList<int> HPB::list_70 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16,
0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x41, 0x42, 0x50, 0x51, 0x60,
0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA2, 0xA3, 0x←
A4, 0xA5, 0xA6, 0xA7, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1,
0xD2, 0xD3, 0xD4, 0xD5, 0xD6, 0xD7, 0xD8, 0xD9, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 }  [inline]
```

5.1.2.18 PTCIDColumn

```
int HPB::PTCIDColumn = 0  [inline], [constexpr]
```

5.1.2.19 PTColumnCount

```
int HPB::PTColumnCount = 9  [inline], [constexpr]
```

5.1.2.20 PTDatasetBaseColumn

```
int HPB::PTDatasetBaseColumn = 2  [inline], [constexpr]
```

5.1.2.21 PTDatasetSpecColumn

```
int HPB::PTDatasetSpecColumn = 3  [inline], [constexpr]
```

5.1.2.22 PTHasSpecificationsColumn

```
int HPB::PTHasSpecificationsColumn = 7  [inline], [constexpr]
```

5.1.2.23 PTIsPromptColumn

```
int HPB::PTIsPromptColumn = 4  [inline], [constexpr]
```

5.1.2.24 PTIsSelectedColumn

```
int HPB::PTIsSelectedColumn = 8  [inline], [constexpr]
```

5.1.2.25 PTNameIDColumn

```
int HPB::PTNameIDColumn = 1  [inline], [constexpr]
```

5.1.2.26 PTPromptContentsColumn

```
int HPB::PTPromptContentsColumn = 5  [inline], [constexpr]
```

5.1.2.27 PTReferencesColumn

```
int HPB::PTReferencesColumn = 6  [inline], [constexpr]
```

5.2 Ui Namespace Reference

Chapter 6

Class Documentation

6.1 BooleanParser Class Reference

```
#include <booleanparser.hpp>
```

Public Member Functions

- BooleanParser ()
- bool parse (const QString &formula, Expression &expr)
- bool check (const QString &formula)

6.1.1 Constructor & Destructor Documentation

6.1.1.1 BooleanParser()

```
BooleanParser::BooleanParser ()
```

6.1.2 Member Function Documentation

6.1.2.1 check()

```
bool BooleanParser::check (
             const QString & formula)
```

6.1.2.2 parse()

```
bool BooleanParser::parse (
             const QString & formula,
             Expression & expr)
```

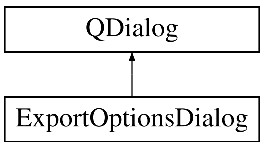
The documentation for this class was generated from the following files:

- src/parser/booleanparser.hpp
- src/parser/booleanparser.cpp

6.2 ExportOptionsDialog Class Reference

```
#include <exportoptionsdialog.hpp>
```

Inheritance diagram for ExportOptionsDialog:



Public Member Functions

- ExportOptionsDialog (QString &outputPath, QString &outputFile, QString &compilationName, QString &helmDataJson, QWidget ∗parent=nullptr)
- ∼ExportOptionsDialog () override

6.2.1 Constructor & Destructor Documentation

6.2.1.1 ExportOptionsDialog()

```
ExportOptionsDialog::ExportOptionsDialog (
            QString & outputPath,
            QString & outputFile,
            QString & compilationName,
            QString & helmDataJson,
            QWidget * parent = nullptr)  [explicit]
```

6.2.1.2 ∼ExportOptionsDialog()

```
ExportOptionsDialog::∼ExportOptionsDialog ()  [override]
```

The documentation for this class was generated from the following files:

- src/dialogs/exportoptionsdialog.hpp
- src/dialogs/exportoptionsdialog.cpp

6.3 Expression Class Reference

```
#include <expression.hpp>
```

Public Member Functions

- Expression ()=default
- Expression (Operator op, QString literal="", const QList< Expression > &={})
- void setOperator (Operator op)
- void addOperand (const Expression &expr)
- void addOperands (const QList< Expression > &expressions)
- const Expression & lhs () const
- const Expression & rhs () const
- const Expression & scope () const
- const QString & literal () const
- Operator op () const
- void clear ()

6.3.1 Constructor & Destructor Documentation

6.3.1.1 Expression() `[1/2]`

```
Expression::Expression ()  [default]
```

6.3.1.2 Expression() `[2/2]`

```
Expression::Expression (
             Operator op,
             QString literal = "",
             const QList< Expression > & children = {})
```

6.3.2 Member Function Documentation

6.3.2.1 addOperand()

```
void Expression::addOperand (
             const Expression & expr)
```

6.3.2.2 addOperands()

```
void Expression::addOperands (
             const QList< Expression > & expressions)
```

6.3.2.3 clear()

```
void Expression::clear ()
```

6.3.2.4 lhs()

```
const Expression & Expression::lhs () const
```

6.3.2.5 literal()

```
const QString & Expression::literal () const
```

6.3.2.6 op()

```
Operator Expression::op () const
```

6.3.2.7 rhs()

```
const Expression & Expression::rhs () const
```

6.3.2.8 scope()

```
const Expression & Expression::scope () const
```

6.3.2.9 setOperator()

```
void Expression::setOperator (
            Operator op)
```

The documentation for this class was generated from the following files:

- src/parser/expression.hpp
- src/parser/expression.cpp

6.4 LanguageModel Class Reference

```
#include <languagemodel.hpp>
```

Public Member Functions

- LanguageModel (int id, QString name, double parameters)
- const QString & name () const
- double parameters () const
- HPB::Vendor vendor () const
- int id () const

6.4.1 Constructor & Destructor Documentation

6.4.1.1 LanguageModel()

```
LanguageModel::LanguageModel (
            int id,
            QString name,
            double parameters)
```

6.4.2 Member Function Documentation

6.4.2.1 id()

```
int LanguageModel::id () const
```

6.4.2.2 name()

```
const QString & LanguageModel::name () const
```

6.4.2.3 parameters()

```
double LanguageModel::parameters () const
```

6.4.2.4 vendor()

```
HPB::Vendor LanguageModel::vendor () const
```

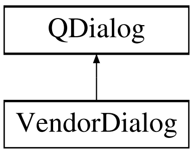
The documentation for this class was generated from the following files:

- src/languagemodel.hpp
- src/languagemodel.cpp

6.5 VendorDialog Class Reference

```
#include <vendordialog.hpp>
```

Inheritance diagram for VendorDialog:



Public Member Functions

- VendorDialog (QList< int > &vendorList, QWidget ∗parent=nullptr)
- ∼VendorDialog () override

6.5.1 Constructor & Destructor Documentation

6.5.1.1 VendorDialog()

```
VendorDialog::VendorDialog (
            QList< int > & vendorList,
            QWidget ∗ parent = nullptr)  [explicit]
```

6.5.1.2 ∼VendorDialog()

```
VendorDialog::∼VendorDialog ()  [override]
```

The documentation for this class was generated from the following files:

- src/dialogs/vendordialog.hpp
- src/dialogs/vendordialog.cpp

Chapter 7

File Documentation

7.1 src/dialogs/exportoptionsdialog.cpp File Reference

```
#include "exportoptionsdialog.hpp"
#include "ui_exportoptionsdialog.h"
#include <QFileDialog>
#include <QMessageBox>
#include <QStandardPaths>
```

7.2 src/dialogs/exportoptionsdialog.hpp File Reference

```
#include <QDialog>
```

Classes

- class ExportOptionsDialog

Namespaces

- namespace Ui

7.3 exportoptionsdialog.hpp

Go to the documentation of this file.

```
#pragma once

#include <QDialog>

namespace Ui {
class ExportOptionsDialog;
} // namespace Ui

class ExportOptionsDialog : public QDialog
{
    Q_OBJECT

public:
    explicit ExportOptionsDialog(QString& outputPath, QString& outputFile, QString& compilationName,
QString& helmDataJson, QWidget *parent = nullptr);
    ~ExportOptionsDialog() override;

private slots:
    void on_buttonBox_accepted();
    void on_export_path_pushButton_clicked();
    void on_helmDataJSON_pushButton_clicked();

private:
    Ui::ExportOptionsDialog *ui;

    QString& m_outputPath;
    QString& m_outputFile;
    QString& m_compilationName;
    QString& m_helmDataJSON;
};
```

7.4 src/dialogs/vendordialog.cpp File Reference

```
#include "vendordialog.hpp"
#include "ui_vendordialog.h"
#include <QListWidget>
#include <QListWidgetItem>
```

7.5 src/dialogs/vendordialog.hpp File Reference

```
#include <QDialog>
#include <QList>
```

Classes

- class VendorDialog

Namespaces

- namespace Ui

7.6 vendordialog.hpp

Go to the documentation of this file.

```
#pragma once

#include <QDialog>
#include <QList>

namespace Ui {
class VendorDialog;
} // namespace Ui

class VendorDialog : public QDialog
{
    Q_OBJECT

public:
    explicit VendorDialog(QList<int>& vendorList, QWidget *parent = nullptr);
    ~VendorDialog() override;

private slots:
    void on_buttonBox_accepted();

    void on_clearAll_pushButton_clicked();

    void on_selectAll_pushButton_clicked();

private:
    Ui::VendorDialog *ui;
    QList<int>& m_VendorList;
};
```

7.7 src/helperfunctions.cpp File Reference

```
#include "helperfunctions.hpp"
#include <QCheckBox>
#include <QDir>
#include <QFile>
#include <QJsonArray>
#include <QJsonDocument>
#include <QList>
#include <QMessageBox>
#include <QRegularExpression>
#include <QString>
#include <QSysInfo>
#include <QTimer>
#include <QTreeWidgetItem>
#include "hpb_globals.hpp"
```

Functions

- int Ask (const QString &text, const QString &informativeText, bool &dontShowAgain)

  Displays a message box with Yes/No options and an optional "Don't show again" checkbox.
- void PopUp (const QString &message)

  Displays a popup message box that automatically closes after 1.5 seconds.
- void Warn (const QString &message)

  Displays a warning message box.
- QJsonObject generateCustomDataset (const QTreeWidgetItem *item, const QString &datasetBase, const QString &datasetSpec, const QJsonObject &helmDataJson)

  Generates a custom dataset JSON object based on tree widget item and dataset specifications.
- QJsonObject getSamples (const QTreeWidgetItem *item)

  Extracts sample data from a QTreeWidgetItem and returns it as a JSON object.
- QJsonDocument getTaskInstances (const QString &taskDir, const QString &helmDataPath)

  Loads task instances from a JSON file within the specified directory.
- QJsonObject loadHelmDataConfig (const QString &helmDataJson)

  Loads the configuration for Helm dataset from a JSON file.
- QString getPromptText (const QJsonObject &obj, const QString &dataset)

  Constructs a formatted prompt text from a JSON object.
- QString getReferencesText (const QJsonObject &obj, const QString &dataset)

  Retrieves formatted references text from a JSON object.

- void addPromptsToTree (const QString &dataset, const QJsonDocument &instances, const QList< QPair< QStringList, QStringList > > &queries, const bool searchIsCaseSensitive, const bool searchIsRegex, QTreeWidget *tree)

  Adds prompts matching search criteria to a QTreeWidget.
- void deleteDatasetFromTree (const QString &datasetName, QTreeWidget *tree)

  Deletes a dataset and its prompts from the QTreeWidget.
- QStringList getFiltersFromDatasetList (const QStringList &datasetNames)

  Generates a list of file filters for dataset directories based on the OS.
- QStringList getHelmTaskDirs (const QStringList &datasets, const QString &helmDataPath)

  Retrieves Helm task directories based on dataset names and path.
- QStringList getSelectedDatasetNames (const QTreeWidget *tree)

  Retrieves the list of selected datasets from a QTreeWidget.
- bool hasSelectedPrompts (const QTreeWidgetItem *item)

  Checks if a QTreeWidgetItem has any selected prompts.
- bool matches (const QString &prompt, const QList< QPair< QStringList, QStringList > > &queries, bool searchIsCaseSensitive, bool searchIsRegex)

  Determines if a given prompt matches any query based on inclusion and exclusion terms.
- QPair< QString, QString > splitDatasetName (const QString &dataset)

  Splits a dataset name into base and specification parts.
- void transformDatasetTree (QTreeWidget *datasetTree, const std::function< void(QTreeWidgetItem *)> &transformation)

  Transforms all dataset entries in a QTreeWidget using a given transformation function.
- void transformPromptTree (QTreeWidget *promptTree, const std::function< void(QTreeWidgetItem *)> &transformation)

  Transforms all prompt entries in a QTreeWidget using a given transformation function.
- QString getCID (const QTreeWidgetItem *item)
- QString getDatasetBase (const QTreeWidgetItem *item)
- QString getDatasetSpec (const QTreeWidgetItem *item)
- QString getName (const QTreeWidgetItem *item)
- QString getPID (const QTreeWidgetItem *item)
- QString getPrompt (const QTreeWidgetItem *item)
- QString getReferences (const QTreeWidgetItem *item)
- bool hasSpecifications (const QTreeWidgetItem *item)
- bool isPrompt (const QTreeWidgetItem *item)
- bool isSelected (const QTreeWidgetItem *item)
- void setCID (QTreeWidgetItem *item, const QString &cid)
- void setSelectedStatus (QTreeWidgetItem *item, bool status)

7.7.1 Function Documentation

7.7.1.1 addPromptsToTree()

```
void addPromptsToTree (
            const QString & dataset,
            const QJsonDocument & instances,
            const QList< QPair< QStringList, QStringList > > & queries,
            const bool searchIsCaseSensitive,
            const bool searchIsRegex,
            QTreeWidget * tree)
```

Adds prompts matching search criteria to a QTreeWidget.

Parameters

dataset	The dataset name.
instances	The JSON document containing instance data.
queries	List of query pairs (inclusions and exclusions).
searchIsCaseSensitive	Boolean flag indicating case-sensitive search.
searchIsRegex	Boolean flag indicating if search terms are regular expressions.
tree	The QTreeWidget to populate with matched prompts.

7.7.1.2 Ask()

```
int Ask (
            const QString & text,
            const QString & informativeText,
            bool & dontShowAgain)
```

Displays a message box with Yes/No options and an optional "Don't show again" checkbox.

Parameters

text	The main text of the message box.
informativeText	Additional information displayed in the message box.
dontShowAgain	Reference to a boolean variable indicating if the "Don't show again" checkbox was checked.

Returns

int The button pressed by the user (QMessageBox::Yes or QMessageBox::No).

7.7.1.3 deleteDatasetFromTree()

```
void deleteDatasetFromTree (
            const QString & datasetName,
            QTreeWidget * tree)
```

Deletes a dataset and its prompts from the QTreeWidget.

Parameters

datasetName	The name of the dataset to delete.
tree	The QTreeWidget containing the dataset structure.

7.7.1.4 generateCustomDataset()

```
QJsonObject generateCustomDataset (
            const QTreeWidgetItem * item,
            const QString & datasetBase,
            const QString & datasetSpec,
            const QJsonObject & helmDataJson)
```

Generates a custom dataset JSON object based on tree widget item and dataset specifications.

Parameters

item	The tree widget item representing a dataset entry.
datasetBase	The base name of the dataset.
datasetSpec	The dataset specification (optional).
helmDataJson	The JSON object containing dataset metadata.

Returns

QJsonObject The generated dataset JSON object.

7.7.1.5 getCID()

```
QString getCID (
            const QTreeWidgetItem * item)
```

7.7.1.6 getDatasetBase()

```
QString getDatasetBase (
            const QTreeWidgetItem * item)
```

7.7.1.7 getDatasetSpec()

```
QString getDatasetSpec (
            const QTreeWidgetItem * item)
```

7.7.1.8 getFiltersFromDatasetList()

```
QStringList getFiltersFromDatasetList (
            const QStringList & datasetNames)
```

Generates a list of file filters for dataset directories based on the OS.

Parameters

datasetNames	The list of dataset names.

Returns

QStringList The list of formatted dataset filters.

7.7.1.9 getHelmTaskDirs()

```
QStringList getHelmTaskDirs (
            const QStringList & datasets,
            const QString & helmDataPath)
```

Retrieves Helm task directories based on dataset names and path.

Parameters

datasets	The list of dataset names.
helmDataPath	The base path for Helm data.

Returns

QStringList The list of task directories.

7.7.1.10 getName()

```
QString getName (
             const QTreeWidgetItem * item)
```

7.7.1.11 getPID()

```
QString getPID (
             const QTreeWidgetItem * item)
```

7.7.1.12 getPrompt()

```
QString getPrompt (
             const QTreeWidgetItem * item)
```

7.7.1.13 getPromptText()

```
QString getPromptText (
             const QJsonObject & obj,
             const QString & dataset)
```

Constructs a formatted prompt text from a JSON object.

Parameters

obj	The JSON object containing prompt details.
dataset	The dataset name associated with the prompt.

Returns

QString The formatted prompt text.

7.7.1.14 getReferences()

```
QString getReferences (
             const QTreeWidgetItem * item)
```

7.7.1.15 getReferencesText()

```
QString getReferencesText (
             const QJsonObject & obj,
             const QString & dataset)
```

Retrieves formatted references text from a JSON object.

Parameters

obj	The JSON object containing reference details.
dataset	The dataset name associated with the references.

Returns

QString The formatted references text.

7.7.1.16 getSamples()

```
QJsonObject getSamples (
            const QTreeWidgetItem * item)
```

Extracts sample data from a QTreeWidgetItem and returns it as a JSON object.

Parameters

item	The tree widget item representing a dataset entry.

Returns

QJsonObject The extracted samples.

7.7.1.17 getSelectedDatasetNames()

```
QStringList getSelectedDatasetNames (
            const QTreeWidget * tree)
```

Retrieves the list of selected datasets from a QTreeWidget.

Parameters

tree	The QTreeWidget representing the dataset structure.

Returns

QStringList The list of selected dataset names.

7.7.1.18 getTaskInstances()

```
QJsonDocument getTaskInstances (
            const QString & taskDir,
            const QString & helmDataPath)
```

Loads task instances from a JSON file within the specified directory.

Parameters

taskDir	The directory containing the instances file.
helmDataPath	The base path for the dataset.

Returns

QJsonDocument The loaded task instances as a JSON document.

7.7.1.19 hasSelectedPrompts()

```
bool hasSelectedPrompts (
            const QTreeWidgetItem * item)
```

Checks if a QTreeWidgetItem has any selected prompts.

Parameters

item	The tree widget item to check.

Returns

bool True if the item has selected prompts, false otherwise.

7.7.1.20 hasSpecifications()

```
bool hasSpecifications (
            const QTreeWidgetItem * item)
```

7.7.1.21 isPrompt()

```
bool isPrompt (
            const QTreeWidgetItem * item)
```

7.7.1.22 isSelected()

```
bool isSelected (
            const QTreeWidgetItem * item)
```

7.7.1.23 loadHelmDataConfig()

```
QJsonObject loadHelmDataConfig (
            const QString & helmDataJson)
```

Loads the configuration for Helm dataset from a JSON file.

Parameters

helmDataJson	Path to the JSON file containing the Helm dataset configuration.

Returns

QJsonObject The parsed JSON object containing the dataset configuration.

7.7.1.24 matches()

```
bool matches (
            const QString & prompt,
            const QList< QPair< QStringList, QStringList > > & queries,
            bool searchIsCaseSensitive,
            bool searchIsRegex)
```

Determines if a given prompt matches any query based on inclusion and exclusion terms.

This function checks if the provided `prompt` satisfies at least one query from `queries`. Each query consists of inclusion and exclusion term lists:

- The prompt must contain all inclusion terms.
- The prompt must not contain any exclusion terms.

The function supports both case-sensitive and case-insensitive searches, as well as regular expression matching.

Parameters

prompt	The text to be matched against the queries.
queries	A list of queries, where each query contains a pair of: • A list of inclusion terms (all must be present). • A list of exclusion terms (none must be present).
searchIsCaseSensitive	If true, the search is case-sensitive; otherwise, it's case-insensitive.
searchIsRegex	If true, terms are treated as regular expressions; otherwise, they are treated as plain text.

Returns

True if the prompt matches at least one query (meeting all inclusions and avoiding all exclusions); otherwise, false.

7.7.1.25 PopUp()

```
void PopUp (
            const QString & message)
```

Displays a popup message box that automatically closes after 1.5 seconds.

Parameters

message	The message to be displayed in the popup.

7.7.1.26 setCID()

```
void setCID (
            QTreeWidgetItem * item,
            const QString & cid)
```

7.7.1.27 setSelectedStatus()

```
void setSelectedStatus (
            QTreeWidgetItem * item,
            bool status)
```

7.7.1.28 splitDatasetName()

```
QPair< QString, QString > splitDatasetName (
            const QString & dataset)
```

Splits a dataset name into base and specification parts.

Parameters

dataset	The dataset name.

Returns

QPair<QString, QString> The separated base name and specification.

7.7.1.29 transformDatasetTree()

```
void transformDatasetTree (
            QTreeWidget * datasetTree,
            const std::function< void(QTreeWidgetItem *)> & transformation)
```

Transforms all dataset entries in a QTreeWidget using a given transformation function.

Parameters

datasetTree	The QTreeWidget representing datasets.
transformation	The transformation function to apply.

7.7.1.30 transformPromptTree()

```
void transformPromptTree (
            QTreeWidget * promptTree,
            const std::function< void(QTreeWidgetItem *)> & transformation)
```

Transforms all prompt entries in a QTreeWidget using a given transformation function.

Parameters

promptTree	The QTreeWidget representing prompts.
transformation	The transformation function to apply.

7.7.1.31 Warn()

```
void Warn (
            const QString & message)
```

Displays a warning message box.

Parameters

message	The warning message to be displayed.

7.8 src/helperfunctions.hpp File Reference

```
#include <functional>
#include <ranges>
#include <QJsonObject>
#include <QString>
#include <QStringList>
#include <QTreeWidget>
#include <QTreeWidgetItem>
```

Functions

- int Ask (const QString &text, const QString &informativeText, bool &dontShowAgain)

  Displays a message box with Yes/No options and an optional "Don't show again" checkbox.
- void PopUp (const QString &message)

  Displays a popup message box that automatically closes after 1.5 seconds.
- void Warn (const QString &message)

  Displays a warning message box.
- QJsonObject generateCustomDataset (const QTreeWidgetItem ∗item, const QString &datasetBase, const QString &datasetSpec, const QJsonObject &helmDataJson)

  Generates a custom dataset JSON object based on tree widget item and dataset specifications.
- QJsonObject getSamples (const QTreeWidgetItem ∗item)

  Extracts sample data from a QTreeWidgetItem and returns it as a JSON object.
- QJsonDocument getTaskInstances (const QString &taskDir, const QString &helmDataPath)

  Loads task instances from a JSON file within the specified directory.
- QJsonObject loadHelmDataConfig (const QString &helmDataJson)

  Loads the configuration for Helm dataset from a JSON file.
- QString prettyPrint (const QJsonObject &obj, const QString &dataset)
- QStringList getFiltersFromDatasetList (const QStringList &datasetNames)

  Generates a list of file filters for dataset directories based on the OS.

- const QList< int > & getModelList (const QTreeWidgetItem ∗)
- QStringList getSelectedDatasetNames (const QTreeWidget ∗tree)

    Retrieves the list of selected datasets from a QTreeWidget.
- void transformDatasetTree (QTreeWidget ∗datasetTree, const std::function< void(QTreeWidgetItem ∗)> &transformation)

    Transforms all dataset entries in a QTreeWidget using a given transformation function.
- void addPromptsToTree (const QString &dataset, const QJsonDocument &instances, const QList< QPair< QStringList, QStringList > > &queries, bool searchIsCaseSensitive, bool searchIsRegex, QTreeWidget ∗tree)

    Adds prompts matching search criteria to a QTreeWidget.
- void deleteDatasetFromTree (const QString &datasetName, QTreeWidget ∗tree)

    Deletes a dataset and its prompts from the QTreeWidget.
- bool hasSelectedPrompts (const QTreeWidgetItem ∗item)

    Checks if a QTreeWidgetItem has any selected prompts.
- void transformPromptTree (QTreeWidget ∗promptTree, const std::function< void(QTreeWidgetItem ∗)> &transformation)

    Transforms all prompt entries in a QTreeWidget using a given transformation function.
- QStringList getHelmTaskDirs (const QStringList &datasets, const QString &helmDataPath)

    Retrieves Helm task directories based on dataset names and path.
- QPair< QString, QString > splitDatasetName (const QString &dataset)

    Splits a dataset name into base and specification parts.
- QString getCID (const QTreeWidgetItem ∗item)
- QString getDatasetBase (const QTreeWidgetItem ∗item)
- QString getDatasetSpec (const QTreeWidgetItem ∗item)
- QString getName (const QTreeWidgetItem ∗item)
- QString getPID (const QTreeWidgetItem ∗item)
- QString getPrompt (const QTreeWidgetItem ∗item)
- QString getReferences (const QTreeWidgetItem ∗item)
- bool hasSpecifications (const QTreeWidgetItem ∗item)
- bool isPrompt (const QTreeWidgetItem ∗item)
- bool isSelected (const QTreeWidgetItem ∗item)
- bool matches (const QString &prompt, const QList< QPair< QStringList, QStringList > > &queries, bool searchIsCaseSensitive, bool searchIsRegex)

    Determines if a given prompt matches any query based on inclusion and exclusion terms.
- void setCID (QTreeWidgetItem ∗item, const QString &cid)
- void setSelectedStatus (QTreeWidgetItem ∗item, bool status)

Variables

- auto _range = [ ] (auto min, auto max) { return std::views::iota(min, max); }

7.8.1 Function Documentation

7.8.1.1 addPromptsToTree()

```
void addPromptsToTree (
             const QString & dataset,
             const QJsonDocument & instances,
             const QList< QPair< QStringList, QStringList > > & queries,
             const bool searchIsCaseSensitive,
             const bool searchIsRegex,
             QTreeWidget ∗ tree)
```

Adds prompts matching search criteria to a QTreeWidget.

Parameters

dataset	The dataset name.
instances	The JSON document containing instance data.
queries	List of query pairs (inclusions and exclusions).
searchIsCaseSensitive	Boolean flag indicating case-sensitive search.
searchIsRegex	Boolean flag indicating if search terms are regular expressions.
tree	The QTreeWidget to populate with matched prompts.

7.8.1.2 Ask()

```
int Ask (
            const QString & text,
            const QString & informativeText,
            bool & dontShowAgain)
```

Displays a message box with Yes/No options and an optional "Don't show again" checkbox.

Parameters

text	The main text of the message box.
informativeText	Additional information displayed in the message box.
dontShowAgain	Reference to a boolean variable indicating if the "Don't show again" checkbox was checked.

Returns

int The button pressed by the user (QMessageBox::Yes or QMessageBox::No).

7.8.1.3 deleteDatasetFromTree()

```
void deleteDatasetFromTree (
            const QString & datasetName,
            QTreeWidget * tree)
```

Deletes a dataset and its prompts from the QTreeWidget.

Parameters

datasetName	The name of the dataset to delete.
tree	The QTreeWidget containing the dataset structure.

7.8.1.4 generateCustomDataset()

```
QJsonObject generateCustomDataset (
            const QTreeWidgetItem * item,
            const QString & datasetBase,
            const QString & datasetSpec,
            const QJsonObject & helmDataJson)
```

Generates a custom dataset JSON object based on tree widget item and dataset specifications.

Parameters

item	The tree widget item representing a dataset entry.
datasetBase	The base name of the dataset.
datasetSpec	The dataset specification (optional).
helmDataJson	The JSON object containing dataset metadata.

Returns

QJsonObject The generated dataset JSON object.

7.8.1.5 getCID()

```
QString getCID (
            const QTreeWidgetItem * item)
```

7.8.1.6 getDatasetBase()

```
QString getDatasetBase (
            const QTreeWidgetItem * item)
```

7.8.1.7 getDatasetSpec()

```
QString getDatasetSpec (
            const QTreeWidgetItem * item)
```

7.8.1.8 getFiltersFromDatasetList()

```
QStringList getFiltersFromDatasetList (
            const QStringList & datasetNames)
```

Generates a list of file filters for dataset directories based on the OS.

Parameters

datasetNames	The list of dataset names.

Returns

QStringList The list of formatted dataset filters.

7.8.1.9 getHelmTaskDirs()

```
QStringList getHelmTaskDirs (
            const QStringList & datasets,
            const QString & helmDataPath)
```

Retrieves Helm task directories based on dataset names and path.

Parameters

datasets	The list of dataset names.
helmDataPath	The base path for Helm data.

Returns

QStringList The list of task directories.

7.8.1.10 getModelList()

```
const QList< int > & getModelList (
            const QTreeWidgetItem * )
```

7.8.1.11 getName()

```
QString getName (
            const QTreeWidgetItem * item)
```

7.8.1.12 getPID()

```
QString getPID (
            const QTreeWidgetItem * item)
```

7.8.1.13 getPrompt()

```
QString getPrompt (
            const QTreeWidgetItem * item)
```

7.8.1.14 getReferences()

```
QString getReferences (
            const QTreeWidgetItem * item)
```

7.8.1.15 getSamples()

```
QJsonObject getSamples (
            const QTreeWidgetItem * item)
```

Extracts sample data from a QTreeWidgetItem and returns it as a JSON object.

Parameters

item	The tree widget item representing a dataset entry.

Returns

QJsonObject The extracted samples.

7.8.1.16 getSelectedDatasetNames()

```
QStringList getSelectedDatasetNames (
            const QTreeWidget * tree)
```

Retrieves the list of selected datasets from a QTreeWidget.

Parameters

tree	The QTreeWidget representing the dataset structure.

Returns

QStringList The list of selected dataset names.

7.8.1.17 getTaskInstances()

```
QJsonDocument getTaskInstances (
            const QString & taskDir,
            const QString & helmDataPath)
```

Loads task instances from a JSON file within the specified directory.

Parameters

taskDir	The directory containing the instances file.
helmDataPath	The base path for the dataset.

Returns

QJsonDocument The loaded task instances as a JSON document.

7.8.1.18 hasSelectedPrompts()

```
bool hasSelectedPrompts (
            const QTreeWidgetItem * item)
```

Checks if a QTreeWidgetItem has any selected prompts.

Parameters

item	The tree widget item to check.

Returns

bool True if the item has selected prompts, false otherwise.

7.8.1.19 hasSpecifications()

```
bool hasSpecifications (
            const QTreeWidgetItem * item)
```

7.8.1.20 isPrompt()

```
bool isPrompt (
            const QTreeWidgetItem * item)
```

7.8.1.21 isSelected()

```
bool isSelected (
            const QTreeWidgetItem * item)
```

7.8.1.22 loadHelmDataConfig()

```
QJsonObject loadHelmDataConfig (
            const QString & helmDataJson)
```

Loads the configuration for Helm dataset from a JSON file.

Parameters

helmDataJson	Path to the JSON file containing the Helm dataset configuration.

Returns

QJsonObject The parsed JSON object containing the dataset configuration.

7.8.1.23 matches()

```
bool matches (
            const QString & prompt,
            const QList< QPair< QStringList, QStringList > > & queries,
            bool searchIsCaseSensitive,
            bool searchIsRegex)
```

Determines if a given prompt matches any query based on inclusion and exclusion terms.

This function checks if the provided `prompt` satisfies at least one query from `queries`. Each query consists of inclusion and exclusion term lists:

- The prompt must contain all inclusion terms.
- The prompt must not contain any exclusion terms.

The function supports both case-sensitive and case-insensitive searches, as well as regular expression matching.

Parameters

prompt	The text to be matched against the queries.
queries	A list of queries, where each query contains a pair of: • A list of inclusion terms (all must be present). • A list of exclusion terms (none must be present).
searchIsCaseSensitive	If true, the search is case-sensitive; otherwise, it's case-insensitive.
searchIsRegex	If true, terms are treated as regular expressions; otherwise, they are treated as plain text.

Returns

True if the prompt matches at least one query (meeting all inclusions and avoiding all exclusions); otherwise, false.

7.8.1.24 PopUp()

```
void PopUp (
             const QString & message)
```

Displays a popup message box that automatically closes after 1.5 seconds.

Parameters

message	The message to be displayed in the popup.

7.8.1.25 prettyPrint()

```
QString prettyPrint (
             const QJsonObject & obj,
             const QString & dataset)
```

7.8.1.26 setCID()

```
void setCID (
             QTreeWidgetItem * item,
             const QString & cid)
```

7.8.1.27 setSelectedStatus()

```
void setSelectedStatus (
             QTreeWidgetItem * item,
             bool status)
```

7.8.1.28 splitDatasetName()

```
QPair< QString, QString > splitDatasetName (
             const QString & dataset)
```

Splits a dataset name into base and specification parts.

Parameters

dataset	The dataset name.

Returns

QPair<QString, QString> The separated base name and specification.

7.8.1.29 transformDatasetTree()

```
void transformDatasetTree (
             QTreeWidget * datasetTree,
             const std::function< void(QTreeWidgetItem *)> & transformation)
```

Transforms all dataset entries in a QTreeWidget using a given transformation function.

Parameters

datasetTree	The QTreeWidget representing datasets.
transformation	The transformation function to apply.

7.8.1.30 transformPromptTree()

```
void transformPromptTree (
            QTreeWidget * promptTree,
            const std::function< void(QTreeWidgetItem *)> & transformation)
```

Transforms all prompt entries in a QTreeWidget using a given transformation function.

Parameters

promptTree	The QTreeWidget representing prompts.
transformation	The transformation function to apply.

7.8.1.31 Warn()

```
void Warn (
            const QString & message)
```

Displays a warning message box.

Parameters

message	The warning message to be displayed.

7.8.2 Variable Documentation

7.8.2.1 _range

```
auto _range = [] (auto min, auto max) { return std::views::iota(min, max); }  [inline]
```

7.9 helperfunctions.hpp

Go to the documentation of this file.

```cpp
#pragma once

#include <functional>
#include <ranges>

#include <QJsonObject>
#include <QString>
#include <QStringList>
#include <QTreeWidget>
#include <QTreeWidgetItem>

inline auto _range = [] (auto min, auto max) { return std::views::iota(min, max); };

/*****************
 * QMessageBoxes *
 *****************/

int Ask(const QString& text, const QString& informativeText, bool& dontShowAgain);
void PopUp(const QString& message);
void Warn(const QString& message);

/*******************************
 * QJson convenience functions *
 *******************************/

QJsonObject generateCustomDataset(const QTreeWidgetItem* item, const QString& datasetBase, const
QString& datasetSpec, const QJsonObject& helmDataJson);
QJsonObject getSamples(const QTreeWidgetItem* item);
QJsonDocument getTaskInstances(const QString& taskDir, const QString& helmDataPath);
QJsonObject loadHelmDataConfig(const QString& helmDataJson);
QString prettyPrint(const QJsonObject& obj, const QString& dataset);

/**************************************
 * Dataset tree convenience functions *
 **************************************/

QStringList getFiltersFromDatasetList(const QStringList& datasetNames);
const QList<int>& getModelList(const QTreeWidgetItem*);
QStringList getSelectedDatasetNames(const QTreeWidget* tree);
void transformDatasetTree(QTreeWidget* datasetTree, const std::function<void(QTreeWidgetItem*)>&
transformation);

/************************************************
 * Prompt and prompt tree convenience functions *
 ************************************************/

void addPromptsToTree(const QString& dataset,
                      const QJsonDocument& instances,
                      const QList<QPair<QStringList, QStringList»& queries,
                      bool searchIsCaseSensitive,
                      bool searchIsRegex,
                      QTreeWidget* tree);
void deleteDatasetFromTree(const QString& datasetName, QTreeWidget* tree);
bool hasSelectedPrompts(const QTreeWidgetItem* item);
void transformPromptTree(QTreeWidget* promptTree, const std::function<void(QTreeWidgetItem*)>&
transformation);

QStringList getHelmTaskDirs(const QStringList& datasets, const QString& helmDataPath);
QPair<QString, QString> splitDatasetName(const QString& dataset);


/****************************
 * Prompt-related functions *
 ****************************/

QString getCID(const QTreeWidgetItem* item);
QString getDatasetBase(const QTreeWidgetItem* item);
QString getDatasetSpec(const QTreeWidgetItem* item);
QString getName(const QTreeWidgetItem* item);
QString getPID(const QTreeWidgetItem* item);
QString getPrompt(const QTreeWidgetItem* item);
QString getReferences(const QTreeWidgetItem* item);
bool hasSpecifications(const QTreeWidgetItem* item);
bool isPrompt(const QTreeWidgetItem* item);
bool isSelected(const QTreeWidgetItem* item);
bool matches(const QString& prompt,
             const QList<QPair<QStringList, QStringList»& queries,
             bool searchIsCaseSensitive,
             bool searchIsRegex);
void setCID(QTreeWidgetItem* item, const QString& cid);
void setSelectedStatus(QTreeWidgetItem* item, bool status);
```

7.10 src/hpb_globals.hpp File Reference

```
#include <QList>
```

Namespaces

- namespace HPB

Enumerations

- enum class HPB::Vendor : uint8_t {
HPB::AlephAlpha = 0x0 , HPB::ai21 = 0x1 , HPB::anthropic = 0x2 , HPB::cohere = 0x3 ,
HPB::eleutherai = 0x4 , HPB::lmsys = 0x5 , HPB::meta = 0x6 , HPB::microsoft = 0x7 ,
HPB::mistralai = 0x8 , HPB::mosaicml = 0x9 , HPB::openai = 0xA , HPB::stanford = 0xB ,
HPB::tiiuae = 0xC , HPB::together = 0xD , HPB::writer_palmyra = 0xE }

Variables

- constexpr int HPB::DTColumnCount = 3
- constexpr int HPB::DTDatasetNameColumn = 0
- constexpr int HPB::DTNumberOfModels = 1
- constexpr int HPB::DTLMListColumn = 2
- constexpr int HPB::PTColumnCount = 9
- constexpr int HPB::PTCIDColumn = 0
- constexpr int HPB::PTNameIDColumn = 1
- constexpr int HPB::PTDatasetBaseColumn = 2
- constexpr int HPB::PTDatasetSpecColumn = 3
- constexpr int HPB::PTIsPromptColumn = 4
- constexpr int HPB::PTPromptContentsColumn = 5
- constexpr int HPB::PTReferencesColumn = 6
- constexpr int HPB::PTHasSpecificationsColumn = 7
- constexpr int HPB::PTIsSelectedColumn = 8
- const QList< int > HPB::list_70 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x41, 0x42, 0x50, 0x51, 0x60, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA2, 0xA3, 0xA4, 0xA5, 0xA6, 0xA7, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xD6, 0xD7, 0xD8, 0xD9, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 }
- const QList< int > HPB::list_69 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x42, 0x50, 0x51, 0x60, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA2, 0xA3, 0xA4, 0xA5, 0xA6, 0xA7, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xD6, 0xD7, 0xD8, 0xD9, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 }
- const QList< int > HPB::list_67 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x42, 0x50, 0x51, 0x60, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA4, 0xA5, 0xA6, 0xA7, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xD6, 0xD7, 0xD8, 0xD9, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 }
- const QList< int > HPB::list_66a = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x42, 0x50, 0x51, 0x60, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA4, 0xA5, 0xA6, 0xA7, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xD6, 0xD7, 0xD8, 0xD9, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0 }

- const QList< int > HPB::list_66b = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x42, 0x50, 0x51, 0x60, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA4, 0xA5, 0xA6, 0xA7, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xD6, 0xD7, 0xD8, 0xD9, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 }
- const QList< int > HPB::list_42 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x70, 0x71, 0xA0, 0xA1, 0xA4, 0xA5, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 }
- const QList< int > HPB::list_40 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x70, 0x71, 0xA0, 0xA1, 0xA4, 0xA5, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xDA, 0xDB, 0xDC, 0xDD }
- const QList< int > HPB::list_39 = { 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x70, 0x71, 0xA0, 0xA1, 0xA2, 0xA3, 0xA4, 0xA5, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xDA, 0xDB, 0xDC, 0xDD }
- const QList< int > HPB::list_32 = { 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x70, 0x71, 0xA0, 0xA1, 0xA4, 0xA5, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xD0, 0xD2, 0xD3, 0xD4, 0xD5 }
- const QList< int > HPB::list_24 = { 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x70, 0x71, 0xA0, 0xA1, 0xA4, 0xA5, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xD0, 0xD2, 0xD3, 0xD4, 0xD5 }
- const QList< int > HPB::list_10 = { 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xDA, 0xDB, 0xDC, 0xDD }
- const QList< int > HPB::list_6 = { 0x20, 0xD0, 0xD2, 0xD3, 0xD4, 0xD5 }
- const QList< int > HPB::list_2 = { 0xA2, 0xA3 }

7.11 hpb_globals.hpp

Go to the documentation of this file.

```
#pragma once

#include <QList>

namespace HPB {
    inline constexpr int DTColumnCount = 3;
    inline constexpr int DTDatasetNameColumn = 0;
    inline constexpr int DTNumberOfModels = 1;
    inline constexpr int DTLMListColumn = 2;

    inline constexpr int PTColumnCount = 9;
    inline constexpr int PTCIDColumn = 0;
    inline constexpr int PTNameIDColumn = 1;
    inline constexpr int PTDatasetBaseColumn = 2;
    inline constexpr int PTDatasetSpecColumn = 3;
    inline constexpr int PTIsPromptColumn = 4;
    inline constexpr int PTPromptContentsColumn = 5;
    inline constexpr int PTReferencesColumn = 6;
    inline constexpr int PTHasSpecificationsColumn = 7;
    inline constexpr int PTIsSelectedColumn = 8;

    inline const QList<int> list_70 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16,
0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x41, 0x42, 0x50, 0x51, 0x60, 0x61, 0x62,
0x63, 0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA2, 0xA3, 0xA4, 0xA5, 0xA6, 0xA7,
0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xD6,
0xD7, 0xD8, 0xD9, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 };
    inline const QList<int> list_69 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16,
0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x42, 0x50, 0x51, 0x60, 0x61, 0x62, 0x63,
0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA2, 0xA3, 0xA4, 0xA5, 0xA6, 0xA7, 0xA8,
0xA9, 0xAA, 0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xD6, 0xD7,
0xD8, 0xD9, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 };
    inline const QList<int> list_67 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16,
0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x42, 0x50, 0x51, 0x60, 0x61, 0x62, 0x63,
0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA4, 0xA5, 0xA6, 0xA7, 0xA8, 0xA9, 0xAA,
0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xD6, 0xD7, 0xD8, 0xD9,
0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 };
    inline const QList<int> list_66a = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16,
0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x42, 0x50, 0x51, 0x60, 0x61, 0x62, 0x63,
0x64, 0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA4, 0xA5, 0xA6, 0xA7, 0xA8, 0xA9, 0xAA,
0xAB, 0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xD6, 0xD7, 0xD8, 0xD9,
0xDA, 0xDB, 0xDC, 0xDD, 0xE0 };
    inline const QList<int> list_66b = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x20,
0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x40, 0x42, 0x50, 0x51, 0x60, 0x61, 0x62, 0x63, 0x64,
0x65, 0x66, 0x70, 0x71, 0x80, 0x90, 0x91, 0xA0, 0xA1, 0xA4, 0xA5, 0xA6, 0xA7, 0xA8, 0xA9, 0xAA, 0xAB,
0xAC, 0xB0, 0xC0, 0xC1, 0xC2, 0xC3, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xD6, 0xD7, 0xD8, 0xD9, 0xDA,
0xDB, 0xDC, 0xDD, 0xE0, 0xE1 };
    inline const QList<int> list_42 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16,
0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x70, 0x71, 0xA0, 0xA1, 0xA4, 0xA5, 0xA8, 0xA9,
0xAA, 0xAB, 0xAC, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xDA, 0xDB, 0xDC, 0xDD, 0xE0, 0xE1 };
    inline const QList<int> list_40 = { 0x00, 0x01, 0x02, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16,
0x20, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x70, 0x71, 0xA0, 0xA1, 0xA4, 0xA5, 0xA8, 0xA9,
0xAA, 0xAB, 0xAC, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xDA, 0xDB, 0xDC, 0xDD };
    inline const QList<int> list_39 = { 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31,
0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x70, 0x71, 0xA0, 0xA1, 0xA2, 0xA3, 0xA4, 0xA5, 0xA8, 0xA9, 0xAA,
0xAB, 0xAC, 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xDA, 0xDB, 0xDC, 0xDD };
    inline const QList<int> list_32 = { 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x30, 0x31,
0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x70, 0x71, 0xA0, 0xA1, 0xA4, 0xA5, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC,
0xD0, 0xD2, 0xD3, 0xD4, 0xD5 };
    inline const QList<int> list_24 = { 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x20, 0x70, 0x71,
0xA0, 0xA1, 0xA4, 0xA5, 0xA8, 0xA9, 0xAA, 0xAB, 0xAC, 0xD0, 0xD2, 0xD3, 0xD4, 0xD5 };
    inline const QList<int> list_10 = { 0xD0, 0xD1, 0xD2, 0xD3, 0xD4, 0xD5, 0xDA, 0xDB, 0xDC, 0xDD };
    inline const QList<int> list_6 = { 0x20, 0xD0, 0xD2, 0xD3, 0xD4, 0xD5 };
    inline const QList<int> list_2 = { 0xA2, 0xA3 };

    enum class Vendor : uint8_t {
        AlephAlpha = 0x0,
        ai21 = 0x1,
        anthropic = 0x2,
        cohere = 0x3,
        eleutherai = 0x4,
        lmsys = 0x5,
        meta = 0x6,
        microsoft = 0x7,
        mistralai = 0x8,
        mosaicml = 0x9,
        openai = 0xA,
        stanford = 0xB,
        tiiuae = 0xC,
        together = 0xD,
        writer_palmyra = 0xE,
    };

} // namespace HPB
```

7.12 src/languagemodel.cpp File Reference

```
#include "languagemodel.hpp"
```

7.13 src/languagemodel.hpp File Reference

```
#include <QString>
#include "hpb_globals.hpp"
```

Classes

- class LanguageModel

7.14 languagemodel.hpp

Go to the documentation of this file.

```
#pragma once

#include <QString>

```

```
#include "hpb_globals.hpp"

class LanguageModel {
public:
    LanguageModel(int id, QString name, double parameters);
    const QString& name() const;
    double parameters() const;
    HPB::Vendor vendor() const;
    int id() const;

private:
    int m_Id;
    QString m_Name;
    double m_Parameters;
};
```

7.15 src/parser/booleanparser.cpp File Reference

```
#include "booleanparser.hpp"
#include <QChar>
#include <QStack>
```

7.16 src/parser/booleanparser.hpp File Reference

```
#include <QList>
#include <QMap>
#include <QPair>
#include <QString>
#include "expression.hpp"
```

Classes

- class BooleanParser

Enumerations

- enum class TokenType : uint8_t {
  START_SYMBOL , END_SYMBOL , LPAREN , RPAREN ,
  AND , OR , NOT , IDENTIFIER ,
  ILLEGAL }

7.16.1 Enumeration Type Documentation

7.16.1.1 TokenType

`enum class TokenType : uint8_t [strong]`

Enumerator

START_SYMBOL	
END_SYMBOL	

Enumerator

LPAREN	
RPAREN	
AND	
OR	
NOT	
IDENTIFIER	
ILLEGAL	

7.17 booleanparser.hpp

Go to the documentation of this file.

```
#pragma once

#include <QList>
#include <QMap>
#include <QPair>
#include <QString>

#include "expression.hpp"

enum class TokenType : uint8_t { START_SYMBOL, END_SYMBOL, LPAREN, RPAREN, AND, OR, NOT, IDENTIFIER,
ILLEGAL };

class BooleanParser {
public:
    BooleanParser();

    bool parse(const QString& formula, Expression& expr);
    bool check(const QString& formula);

private:
    void advance();
    bool match(TokenType type);
    bool sentence(Expression& expr);
    bool disjunction(Expression& expr);
    bool conjunction(Expression& expr);
    bool negation(Expression& expr);

    void tokenize(const QString& formula);

    int m_Index;
    TokenType m_Sym;
    QList<QPair<QString, TokenType» m_TokenList;

    inline static const QMap<TokenType, QString> TokenTypeName = {
        { TokenType::START_SYMBOL, QString("<S>") },
        { TokenType::END_SYMBOL, QString("<E>") },
        { TokenType::LPAREN, QString("lparen") },
        { TokenType::RPAREN, QString("rparen") },
        { TokenType::NOT, QString("not") },
        { TokenType::AND, QString("and") },
        { TokenType::OR, QString("or") },
        { TokenType::ILLEGAL, QString("illegal") },
    };
};
```

7.18 src/parser/expression.cpp File Reference

```
#include "expression.hpp"
```

7.19 src/parser/expression.hpp File Reference

```
#include <memory>
#include <QList>
#include <QString>
```

Classes

- class Expression

Enumerations

- enum class Operator : uint8_t { NOT , AND , OR , NIL }

7.19.1 Enumeration Type Documentation

7.19.1.1 Operator

```
enum class Operator :  uint8_t  [strong]
```

Enumerator

NOT	
AND	
OR	
NIL	

7.20 expression.hpp

Go to the documentation of this file.

```
#pragma once

#include <memory>

#include <QList>
#include <QString>

enum class Operator : uint8_t { NOT, AND, OR, NIL };

struct Conjunction;
struct Disjunction;
struct Negation;

class Expression
{
public:
    Expression() = default;
    Expression(Operator op, QString literal = "", const QList<Expression>& = {});
    void setOperator(Operator op);
    void addOperand(const Expression& expr);
    void addOperands(const QList<Expression>& expressions);
    const Expression& lhs() const;
    const Expression& rhs() const;
    const Expression& scope() const;
    const QString& literal() const;
    Operator op() const;
    void clear();

private:
    Operator m_Operator = Operator::NIL;
    QString m_Literal = "";
    QList<std::shared_ptr<Expression>> m_Children = {};
};
```

7.21 src/parser/logic.cpp File Reference

```
#include "logic.hpp"
```

Functions

- bool isAtomic (const Expression &expr)
- bool isConjunction (const Expression &expr)
- bool isDisjunction (const Expression &expr)
- bool isNegation (const Expression &expr)
- bool isDNF (const Expression &expr)
- bool isNNF (const Expression &expr)
- Expression NNFtoDNF (const Expression &expr)
- Expression toDNF (const Expression &expr)
- Expression toNNF (const Expression &expr)

7.21.1 Function Documentation

7.21.1.1 isAtomic()

```
bool isAtomic (
          const Expression & expr)
```

7.21.1.2 isConjunction()

```
bool isConjunction (
          const Expression & expr)
```

7.21.1.3 isDisjunction()

```
bool isDisjunction (
          const Expression & expr)
```

7.21.1.4 isDNF()

```
bool isDNF (
          const Expression & expr)
```

7.21.1.5 isNegation()

```
bool isNegation (
          const Expression & expr)
```

7.21.1.6 isNNF()

```
bool isNNF (
            const Expression & expr)
```

7.21.1.7 NNFtoDNF()

```
Expression NNFtoDNF (
            const Expression & expr)
```

7.21.1.8 toDNF()

```
Expression toDNF (
            const Expression & expr)
```

7.21.1.9 toNNF()

```
Expression toNNF (
            const Expression & expr)
```

7.22 src/parser/logic.hpp File Reference

```
#include "expression.hpp"
```

Functions

- bool isNegation (const Expression &expr)
- bool isDisjunction (const Expression &expr)
- bool isConjunction (const Expression &expr)
- bool isAtomic (const Expression &expr)
- bool isNNF (const Expression &expr)
- bool isDNF (const Expression &expr)
- Expression toDNF (const Expression &expr)
- Expression toNNF (const Expression &expr)
- Expression NNFtoDNF (const Expression &expr)

7.22.1 Function Documentation

7.22.1.1 isAtomic()

```
bool isAtomic (
            const Expression & expr)
```

7.22.1.2 isConjunction()

```
bool isConjunction (
             const Expression & expr)
```

7.22.1.3 isDisjunction()

```
bool isDisjunction (
             const Expression & expr)
```

7.22.1.4 isDNF()

```
bool isDNF (
             const Expression & expr)
```

7.22.1.5 isNegation()

```
bool isNegation (
             const Expression & expr)
```

7.22.1.6 isNNF()

```
bool isNNF (
             const Expression & expr)
```

7.22.1.7 NNFtoDNF()

```
Expression NNFtoDNF (
             const Expression & expr)
```

7.22.1.8 toDNF()

```
Expression toDNF (
             const Expression & expr)
```

7.22.1.9 toNNF()

```
Expression toNNF (
             const Expression & expr)
```

7.23 logic.hpp

Go to the documentation of this file.

```
#pragma once

#include "expression.hpp"

bool isNegation(const Expression& expr);
bool isDisjunction(const Expression& expr);
bool isConjunction(const Expression& expr);
bool isAtomic(const Expression& expr);

bool isNNF(const Expression& expr);
bool isDNF(const Expression& expr);

Expression toDNF(const Expression& expr);
Expression toNNF(const Expression& expr);

Expression NNFtoDNF(const Expression& expr);
```

7.24 src/parser/queryparser.cpp File Reference

```
#include "queryparser.hpp"
#include <algorithm>
#include "booleanparser.hpp"
#include "expression.hpp"
#include "logic.hpp"
```

Functions

- bool checkQuery (const QString &query)
- QList< QPair< QStringList, QStringList > > getQueries (const QString &queryStr)

7.24.1 Function Documentation

7.24.1.1 checkQuery()

```
bool checkQuery (
            const QString & query)
```

7.24.1.2 getQueries()

```
QList< QPair< QStringList, QStringList > > getQueries (
            const QString & queryStr)
```

7.25 src/parser/queryparser.hpp File Reference

```
#include <QList>
#include <QPair>
#include <QString>
#include <QStringList>
```

Functions

- bool checkQuery (const QString &queryStr)
- QList< QPair< QStringList, QStringList > > getQueries (const QString &queryStr)

7.25.1 Function Documentation

7.25.1.1 checkQuery()

```
bool checkQuery (
              const QString & queryStr)
```

7.25.1.2 getQueries()

```
QList< QPair< QStringList, QStringList > > getQueries (
              const QString & queryStr)
```

7.26 queryparser.hpp

Go to the documentation of this file.

```
#pragma once

#include <QList>
#include <QPair>
#include <QString>
#include <QStringList>

bool checkQuery(const QString& queryStr);
QList<QPair<QStringList, QStringList» getQueries(const QString& queryStr);
```

Index