

# Documentação de Código-fonte com a Ferramenta **Doxygen**

Alberto Xavier Pavim

Laboratório de Metrologia e Automatização – LABMETRO/EMC  
Sistemas Industriais Inteligentes – S2i/DAS  
Universidade Federal de Santa Catarina

Florianópolis, 02 de Junho de 2006.



# Sumário

- 1 **Introdução**
  - Importância da Documentação do Código-fonte
- 2 Documentando o Código-fonte com Doxygen
  - Estilos de Documentação
- 3 Utilização e Configuração de um Projeto Doxygen
  - A Ferramenta Doxygen
  - A Ferramenta Doxywizard
- 4 Resultado

# Sumário

- 1 Introdução
  - Importância da Documentação do Código-fonte
- 2 Documentando o Código-fonte com Doxygen
  - Estilos de Documentação
- 3 Utilização e Configuração de um Projeto Doxygen
  - A Ferramenta Doxygen
  - A Ferramenta Doxywizard
- 4 Resultado

# Sumário

- 1 Introdução
  - Importância da Documentação do Código-fonte
- 2 Documentando o Código-fonte com Doxygen
  - Estilos de Documentação
- 3 Utilização e Configuração de um Projeto Doxygen
  - A Ferramenta Doxygen
  - A Ferramenta Doxywizard
- 4 Resultado

# Sumário

- 1 Introdução
  - Importância da Documentação do Código-fonte
- 2 Documentando o Código-fonte com Doxygen
  - Estilos de Documentação
- 3 Utilização e Configuração de um Projeto Doxygen
  - A Ferramenta Doxygen
  - A Ferramenta Doxywizard
- 4 Resultado

# Sumário da Subseção

- 1 **Introdução**
  - Importância da Documentação do Código-fonte
- 2 Documentando o Código-fonte com Doxygen
  - Estilos de Documentação
- 3 Utilização e Configuração de um Projeto Doxygen
  - A Ferramenta Doxygen
  - A Ferramenta Doxywizard
- 4 Resultado

# Por que documentar o código-fonte?

Facilitar a **reutilização** e **manutenção** do código-fonte

Disponibilizar documentação profissional de bibliotecas para usuários do *software*

Contribuir com a gestão do conhecimento

# Por que documentar o código-fonte?

Facilitar a **reutilização** e **manutenção** do código-fonte

Disponibilizar documentação profissional de bibliotecas para usuários do *software*

Contribuir com a gestão do conhecimento



# Por que documentar o código-fonte?

Facilitar a **reutilização** e **manutenção** do código-fonte

Disponibilizar documentação profissional de bibliotecas para usuários do *software*

Contribuir com a gestão do conhecimento

# O que é o Doxygen?

É um sistema flexível de documentação de código-fonte, multi-linguagem, multi-plataforma e com múltiplas saídas.

## Linguagens Suportadas

C++, C, Java, Objective-C, Python, IDL, PHP, C# e D.

## Plataformas Suportadas

GNU/Linux (Unix), Mac OS X, Windows.

## Saídas Suportadas

HTML, Latex, RTF, PostScript, PDF, XML, compressed HTML (CHM), Unix man pages.

# O que é o Doxygen?

É um sistema flexível de documentação de código-fonte, multi-linguagem, multi-plataforma e com múltiplas saídas.

## Linguagens Suportadas

C++, C, Java, Objective-C, Python, IDL, PHP, C# e D.

## Plataformas Suportadas

GNU/Linux (Unix), Mac OS X, Windows.

## Saídas Suportadas

HTML, Latex, RTF, PostScript, PDF, XML, compressed HTML (CHM), Unix man pages.

# O que é o Doxygen?

É um sistema flexível de documentação de código-fonte, multi-linguagem, multi-plataforma e com múltiplas saídas.

## Linguagens Suportadas

C++, C, Java, Objective-C, Python, IDL, PHP, C# e D.

## Plataformas Suportadas

GNU/Linux (Unix), Mac OS X, Windows.

## Saídas Suportadas

HTML, Latex, RTF, PostScript, PDF, XML, compressed HTML (CHM), Unix man pages.

# O que é o Doxygen?

É um sistema flexível de documentação de código-fonte, multi-linguagem, multi-plataforma e com múltiplas saídas.

## Linguagens Suportadas

C++, C, Java, Objective-C, Python, IDL, PHP, C# e D.

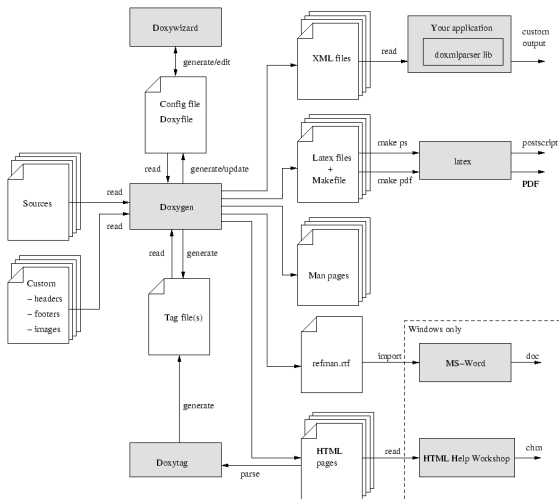
## Plataformas Suportadas

GNU/Linux (Unix), Mac OS X, Windows.

## Saídas Suportadas

HTML, Latex, RTF, PostScript, PDF, XML, compressed HTML (CHM), Unix man pages.

# Arquitetura do Doxygen



# Fontes de Informação para gerar Documentação

Doxygen busca por **tags** de documentação no código-fonte (caso: novos projetos)

É capaz de extrair informação de códigos-fonte não documentados (caso: antigos projetos)

Gera automaticamente diversos gráficos UML de acordo com o relacionamento das entidades envolvidas

É possível escrever documentação comum e ligá-la à documentação do doxygen

# Fontes de Informação para gerar Documentação

Doxygen busca por **tags** de documentação no código-fonte (caso: novos projetos)

É capaz de extrair informação de códigos-fonte não documentados (caso: antigos projetos)

Gera automaticamente diversos gráficos UML de acordo com o relacionamento das entidades envolvidas

É possível escrever documentação comum e ligá-la à documentação do doxygen



# Fontes de Informação para gerar Documentação

Doxygen busca por **tags** de documentação no código-fonte (caso: novos projetos)

É capaz de extrair informação de códigos-fonte não documentados (caso: antigos projetos)

Gera automaticamente diversos gráficos UML de acordo com o relacionamento das entidades envolvidas

É possível escrever documentação comum e ligá-la à documentação do doxygen

# Fontes de Informação para gerar Documentação

Doxygen busca por **tags** de documentação no código-fonte (caso: novos projetos)

É capaz de extrair informação de códigos-fonte não documentados (caso: antigos projetos)

Gera automaticamente diversos gráficos UML de acordo com o relacionamento das entidades envolvidas

É possível escrever documentação comum e ligá-la à documentação do doxygen

# Sumário da Subseção

- 1 Introdução
  - Importância da Documentação do Código-fonte
- 2 Documentando o Código-fonte com Doxygen
  - Estilos de Documentação
- 3 Utilização e Configuração de um Projeto Doxygen
  - A Ferramenta Doxygen
  - A Ferramenta Doxywizard
- 4 Resultado

# Aprendendo com um Exemplo Inicial

```
/// Descricao simples da classe (uma linha)
/**
 * Descricao detalhada da classe.
 * Esta pode ter varias linhas.
 */
class Example
{
    /// Documentacao antes do membro
    int numInt;

    double numDouble; /*!< Documentacao depois do membro */

    /** Descricao simples da funcao ateh o ponto final.

        Descricao detalhada da funcao , podendo ter
        varias linhas dentro deste bloco

        \param a_fFloat Descricao do parametro float

        \return Descricao do valor de retorno
    */
    char * Print( float a_fFloat );
};
```

## Estilos mais Comuns: junto às Entidades

```
//! Documentação simples de uma linha (antes)  
/*! ... */ Documentação de múltiplas linhas (antes)  
//!< Documentação simples de uma linha (depois)  
/*!< ... */ Documentação de múltiplas linhas (depois)  
///  
/** ... */ Mesmo comportamento de /*! ... */  
///  
/**< ... */ Mesmo comportamento de /*!< ... */
```

Prefere-se o uso de `///` e `/** ... */` para manter compatibilidade com estilo de documentação usada em C#

Em Python utiliza-se `##` ao invés de somente `#`

## Estilos mais Comuns: junto às Entidades

```
    //! Documentação simples de uma linha (antes)  
    /*! ... */ Documentação de múltiplas linhas (antes)  
    //!< Documentação simples de uma linha (depois)  
    /*!< ... */ Documentação de múltiplas linhas (depois)  
    /// Mesmo comportamento de //!  
    /** ... */ Mesmo comportamento de /*! ... */  
    ///< Mesmo comportatmento de //!<  
    /**< ... */ Mesmo comportamento de /*!< ... */
```

Prefere-se o uso de `///` e `/** ... */` para manter compatibilidade com estilo de documentação usada em C#

Em Python utiliza-se `##` ao invés de somente `#`

## Estilos mais Comuns: junto às Entidades

```
    //! Documentação simples de uma linha (antes)  
    /*! ... */ Documentação de múltiplas linhas (antes)  
    //!< Documentação simples de uma linha (depois)  
    /*!< ... */ Documentação de múltiplas linhas (depois)  
    /// Mesmo comportamento de //!  
    /** ... */ Mesmo comportamento de /*! ... */  
    ///< Mesmo comportatmento de //!<  
    /**< ... */ Mesmo comportamento de /*!< ... */
```

Prefere-se o uso de `///` e `/** ... */` para manter compatibilidade com estilo de documentação usada em C#

Em Python utiliza-se `##` ao invés de somente `#`

## Estilos mais Comuns: junto às Entidades

```
    //! Documentação simples de uma linha (antes)  
    /*! ... */ Documentação de múltiplas linhas (antes)  
    //!< Documentação simples de uma linha (depois)  
    /*!< ... */ Documentação de múltiplas linhas (depois)  
    /// Mesmo comportamento de //!  
    /** ... */ Mesmo comportamento de /*! ... */  
    ///< Mesmo comportatmento de //!<  
    /**< ... */ Mesmo comportamento de /*!< ... */
```

Prefere-se o uso de `///` e `/** ... */` para manter compatibilidade com estilo de documentação usada em C#

Em Python utiliza-se `##` ao invés de somente `#`



## Estilos mais Comuns: junto às Entidades

```
    //! Documentação simples de uma linha (antes)  
    /*! ... */ Documentação de múltiplas linhas (antes)  
    //!< Documentação simples de uma linha (depois)  
    /*!< ... */ Documentação de múltiplas linhas (depois)  
    /// Mesmo comportamento de //!  
    /** ... */ Mesmo comportamento de /*! ... */  
    ///< Mesmo comportatmento de //!<  
    /**< ... */ Mesmo comportamento de /*!< ... */
```

Prefere-se o uso de `///` e `/** ... */` para manter compatibilidade com estilo de documentação usada em C#

Em Python utiliza-se `##` ao invés de somente `#`

## Estilos mais Comuns: junto às Entidades

`//!` Documentação simples de uma linha (**antes**)  
`/*! ... */` Documentação de múltiplas linhas (**antes**)  
`/*!<` Documentação simples de uma linha (**depois**)  
`/*!< ... */` Documentação de múltiplas linhas (**depois**)  
`///` Mesmo comportamento de `//!`  
`/** ... */` Mesmo comportamento de `/*! ... */`  
`///<` Mesmo comportatmento de `/*!<`  
`/**< ... */` Mesmo comportamento de `/*!< ... */`

Prefere-se o uso de `///` e `/** ... */` para manter compatibilidade com estilo de documentação usada em C#

Em Python utiliza-se `##` ao invés de somente `#`

## Estilos mais Comuns: junto às Entidades

`//!` Documentação simples de uma linha (**antes**)  
`/*! ... */` Documentação de múltiplas linhas (**antes**)  
`/*!<` Documentação simples de uma linha (**depois**)  
`/*!< ... */` Documentação de múltiplas linhas (**depois**)  
`///` Mesmo comportamento de `//!`  
`/** ... */` Mesmo comportamento de `/*! ... */`  
`///<` Mesmo comportamento de `/*!<`  
`/**< ... */` Mesmo comportamento de `/*!< ... */`

Prefere-se o uso de `///` e `/** ... */` para manter compatibilidade com estilo de documentação usada em C#

Em Python utiliza-se `##` ao invés de somente `#`

## Estilos mais Comuns: junto às Entidades

```
    //! Documentação simples de uma linha (antes)  
    /*! ... */ Documentação de múltiplas linhas (antes)  
    //!< Documentação simples de uma linha (depois)  
    /*!< ... */ Documentação de múltiplas linhas (depois)  
    /// Mesmo comportamento de //!  
    /** ... */ Mesmo comportamento de /*! ... */  
    ///< Mesmo comportatmento de //!<  
    /**< ... */ Mesmo comportamento de /*!< ... */
```

Prefere-se o uso de `///` e `/** ... */` para manter compatibilidade com estilo de documentação usada em C#

Em Python utiliza-se `##` ao invés de somente `#`

## Estilos mais Comuns: junto às Entidades

```
    //! Documentação simples de uma linha (antes)  
    /*! ... */ Documentação de múltiplas linhas (antes)  
    //!< Documentação simples de uma linha (depois)  
    /*!< ... */ Documentação de múltiplas linhas (depois)  
    /// Mesmo comportamento de //!  
    /** ... */ Mesmo comportamento de /*! ... */  
    ///< Mesmo comportatmento de //!<  
    /**< ... */ Mesmo comportamento de /*!< ... */
```

Prefere-se o uso de `///` e `/** ... */` para manter compatibilidade com estilo de documentação usada em C#

Em Python utiliza-se `##` ao invés de somente `#`

## Estilos mais Comuns: junto às Entidades

```
    //! Documentação simples de uma linha (antes)  
    /*! ... */ Documentação de múltiplas linhas (antes)  
    //!< Documentação simples de uma linha (depois)  
    /*!< ... */ Documentação de múltiplas linhas (depois)  
    /// Mesmo comportamento de //!  
    /** ... */ Mesmo comportamento de /*! ... */  
    ///< Mesmo comportatmento de //!<  
    /**< ... */ Mesmo comportamento de /*!< ... */
```

Prefere-se o uso de `///` e `/** ... */` para manter compatibilidade com estilo de documentação usada em C#

Em Python utiliza-se `##` ao invés de somente `#`

# Blocos de Documentação Auxiliar

`\brief, @brief` Uma linha simples de documentação

`\param, @param` Parâmetro de uma função

`\return, @return` Valor de retorno de uma função

`\sa, @sa` Documentação auxiliar aconselhada (*see also*)

`\warning, @warning` Documentação de avisos ao usuário

Em Python estes comandos podem ser passados dentro de ***docstrings***

# Blocos de Documentação Auxiliar

`\brief, @brief` Uma linha simples de documentação

`\param, @param` Parâmetro de uma função

`\return, @return` Valor de retorno de uma função

`\sa, @sa` Documentação auxiliar aconselhada (*see also*)

`\warning, @warning` Documentação de avisos ao usuário

Em Python estes comandos podem ser passados dentro de *docstrings*



# Blocos de Documentação Auxiliar

`\brief, @brief` Uma linha simples de documentação

`\param, @param` Parâmetro de uma função

`\return, @return` Valor de retorno de uma função

`\sa, @sa` Documentação auxiliar aconselhada (*see also*)

`\warning, @warning` Documentação de avisos ao usuário

Em Python estes comandos podem ser passados dentro de *docstrings*

# Blocos de Documentação Auxiliar

`\brief, @brief` Uma linha simples de documentação

`\param, @param` Parâmetro de uma função

`\return, @return` Valor de retorno de uma função

`\sa, @sa` Documentação auxiliar aconselhada (*see also*)

`\warning, @warning` Documentação de avisos ao usuário

Em Python estes comandos podem ser passados dentro de *docstrings*

# Blocos de Documentação Auxiliar

`\brief`, `@brief` Uma linha simples de documentação  
`\param`, `@param` Parâmetro de uma função  
`\return`, `@return` Valor de retorno de uma função  
`\sa`, `@sa` Documentação auxiliar aconselhada (*see also*)  
`\warning`, `@warning` Documentação de avisos ao usuário

Em Python estes comandos podem ser passados dentro de *docstrings*

# Blocos de Documentação Auxiliar

`\brief, @brief` Uma linha simples de documentação  
`\param, @param` Parâmetro de uma função  
`\return, @return` Valor de retorno de uma função  
`\sa, @sa` Documentação auxiliar aconselhada (*see also*)  
`\warning, @warning` Documentação de avisos ao usuário

Em Python estes comandos podem ser passados dentro de ***docstrings***

# Estilos Especiais: distante das Entidades

`\class, @class` Classes  
`\struct, @struct` Estruturas  
`\union, @union` Uniões  
`\enum, @enum` Enumerações  
`\fn, @fn` Funções  
`\var, @var` Variáveis  
`\def, @def` #define  
`\file, @file` Arquivos  
`\namespace, @namespace` Namespaces  
`\package, @package` Pacotes  
`\interface, @interface` Interfaces

# Estilos Especiais: distante das Entidades

`\class, @class` Classes

`\struct, @struct` Estruturas

`\union, @union` Uniões

`\enum, @enum` Enumerações

`\fn, @fn` Funções

`\var, @var` Variáveis

`\def, @def` `#define`

`\file, @file` Arquivos

`\namespace, @namespace` Namespaces

`\package, @package` Pacotes

`\interface, @interface` Interfaces

# Estilos Especiais: distante das Entidades

`\class, @class` Classes

`\struct, @struct` Estruturas

`\union, @union` Uniões

`\enum, @enum` Enumerações

`\fn, @fn` Funções

`\var, @var` Variáveis

`\def, @def` `#define`

`\file, @file` Arquivos

`\namespace, @namespace` Namespaces

`\package, @package` Pacotes

`\interface, @interface` Interfaces

# Estilos Especiais: distante das Entidades

`\class, @class` Classes

`\struct, @struct` Estruturas

`\union, @union` Uniões

`\enum, @enum` Enumerações

`\fn, @fn` Funções

`\var, @var` Variáveis

`\def, @def` `#define`

`\file, @file` Arquivos

`\namespace, @namespace` Namespaces

`\package, @package` Pacotes

`\interface, @interface` Interfaces



# Estilos Especiais: distante das Entidades

`\class, @class` Classes

`\struct, @struct` Estruturas

`\union, @union` Uniões

`\enum, @enum` Enumerações

`\fn, @fn` Funções

`\var, @var` Variáveis

`\def, @def` `#define`

`\file, @file` Arquivos

`\namespace, @namespace` Namespaces

`\package, @package` Pacotes

`\interface, @interface` Interfaces

# Estilos Especiais: distante das Entidades

`\class, @class` Classes

`\struct, @struct` Estruturas

`\union, @union` Uniões

`\enum, @enum` Enumerações

`\fn, @fn` Funções

`\var, @var` Variáveis

`\def, @def` `#define`

`\file, @file` Arquivos

`\namespace, @namespace` Namespaces

`\package, @package` Pacotes

`\interface, @interface` Interfaces

# Estilos Especiais: distante das Entidades

`\class, @class` Classes

`\struct, @struct` Estruturas

`\union, @union` Uniões

`\enum, @enum` Enumerações

`\fn, @fn` Funções

`\var, @var` Variáveis

`\def, @def` `#define`

`\file, @file` Arquivos

`\namespace, @namespace` Namespaces

`\package, @package` Pacotes

`\interface, @interface` Interfaces

# Estilos Especiais: distante das Entidades

`\class, @class` Classes

`\struct, @struct` Estruturas

`\union, @union` Uniões

`\enum, @enum` Enumerações

`\fn, @fn` Funções

`\var, @var` Variáveis

`\def, @def` `#define`

`\file, @file` Arquivos

`\namespace, @namespace` Namespaces

`\package, @package` Pacotes

`\interface, @interface` Interfaces

# Estilos Especiais: distante das Entidades

`\class, @class` Classes  
`\struct, @struct` Estruturas  
`\union, @union` Uniões  
`\enum, @enum` Enumerações  
    `\fn, @fn` Funções  
`\var, @var` Variáveis  
`\def, @def` `#define`  
`\file, @file` Arquivos  
`\namespace, @namespace` Namespaces  
`\package, @package` Pacotes  
`\interface, @interface` Interfaces

# Estilos Especiais: distante das Entidades

`\class, @class` Classes  
`\struct, @struct` Estruturas  
`\union, @union` Uniões  
`\enum, @enum` Enumerações  
    `\fn, @fn` Funções  
`\var, @var` Variáveis  
`\def, @def` `#define`  
`\file, @file` Arquivos  
`\namespace, @namespace` Namespaces  
`\package, @package` Pacotes  
`\interface, @interface` Interfaces

# Estilos Especiais: distante das Entidades

`\class, @class` Classes  
`\struct, @struct` Estruturas  
`\union, @union` Uniões  
`\enum, @enum` Enumerações  
    `\fn, @fn` Funções  
`\var, @var` Variáveis  
`\def, @def` `#define`  
`\file, @file` Arquivos  
`\namespace, @namespace` Namespaces  
`\package, @package` Pacotes  
`\interface, @interface` Interfaces

# Um Exemplo mais Elaborado

```
/**
 * \file Arquivo.cpp
 * \brief Pequena descricao do arquivo
 *
 * Descricao mais detalhada do arquivo.
 * Em geral tem mais de uma linha.
 *
 * \sa Arquivo.h
 */

/** @fn int funcao_global( char * pointer )
    @brief Curta explicacao da funcao

    Explicacao detalhada da funcao.

    @param pointer Explicacao do parametro

    @return Explicacao do valor de retorno

    @warning Funcao ainda em desenvolvimento!
 */
int funcao_global( char * pointer )
{
    ...
}
```



# Um Exemplo em Python

```
##
## \file Arquivo.py
## \brief Pequena descricao do arquivo
## \sa OutroArquivo.py
##

# Declaracao da classe
class ExemploPython :
    """Breve comentario da classe termina no ponto.

    Demais comentarios podem vir em mais de uma
    linha, como este aqui.
    """

# -----

def funcao_membro( a_sPointer ) :
    """Curta explicacao da funcao termina neste ponto.

    Explicacao mais detalhada da funcao pode se
    alongar em mais linhas.

    \param a_sPointer Explicacao do argumento.
    \return Explicacao do valor de retorno.
    \warning Funcao inacabada!
    """
```

# Blocos de Documentação Avançada

**Listas** Introdução de listas pontuadas ou enumeradas na documentação

**Grupos** Agrupamento de entidades para melhorar organização da documentação

**Equações** Inclusão de equações junto à documentação

**Diagramas** Geração de grafos e diagramas UML de acordo com o relacionamento das entidades no código

Ler a documentação detalhada do Doxygen:

<http://www.stack.nl/~dimitri/doxygen/manual.html>

# Blocos de Documentação Avançada

**Listas** Introdução de listas pontuadas ou enumeradas na documentação

**Grupos** Agrupamento de entidades para melhorar organização da documentação

**Equações** Inclusão de equações junto à documentação

**Diagramas** Geração de grafos e diagramas UML de acordo com o relacionamento das entidades no código

Ler a documentação detalhada do Doxygen:

<http://www.stack.nl/~dimitri/doxygen/manual.html>

# Blocos de Documentação Avançada

**Listas** Introdução de listas pontuadas ou enumeradas na documentação

**Grupos** Agrupamento de entidades para melhorar organização da documentação

**Equações** Inclusão de equações junto à documentação

**Diagramas** Geração de grafos e diagramas UML de acordo com o relacionamento das entidades no código

Ler a documentação detalhada do Doxygen:

<http://www.stack.nl/~dimitri/doxygen/manual.html>

# Blocos de Documentação Avançada

**Listas** Introdução de listas pontuadas ou enumeradas na documentação

**Grupos** Agrupamento de entidades para melhorar organização da documentação

**Equações** Inclusão de equações junto à documentação

**Diagramas** Geração de grafos e diagramas UML de acordo com o relacionamento das entidades no código

Ler a documentação detalhada do Doxygen:

<http://www.stack.nl/~dimitri/doxygen/manual.html>

# Blocos de Documentação Avançada

**Listas** Introdução de listas pontuadas ou enumeradas na documentação

**Grupos** Agrupamento de entidades para melhorar organização da documentação

**Equações** Inclusão de equações junto à documentação

**Diagramas** Geração de grafos e diagramas UML de acordo com o relacionamento das entidades no código

Ler a documentação detalhada do Doxygen:

<http://www.stack.nl/~dimitri/doxygen/manual.html>

# Sumário da Subseção

- 1 Introdução
  - Importância da Documentação do Código-fonte
- 2 Documentando o Código-fonte com Doxygen
  - Estilos de Documentação
- 3 Utilização e Configuração de um Projeto Doxygen
  - A Ferramenta Doxygen
  - A Ferramenta Doxywizard
- 4 Resultado

# Doxygen na Linha de Comando

```
$ doxygen --help
```

- 1) Use o doxygen para gerar um arquivo template de configuracao:  
`$ doxygen [-s] -g [configName]`
- 2) Use o doxygen para atualizar um arquivo de configuracao antigo:  
`$ doxygen [-s] -u [configName]`
- 3) Use o doxygen para gerar a documentacao atraves de um arquivo de configuracao existente:  
`$ doxygen [configName]`
- 4) Use o doxygen para gerar um arquivo de estilos template para os formatos RTF, HTML e Latex.  
RTF: `$ doxygen -w rtf styleSheetFile`  
HTML: `$ doxygen -w html headerFile footerFile styleSheetFile [configFile]`  
LaTeX: `$ doxygen -w latex headerFile styleSheetFile [configFile]`
- 5) Use o doxygen para gerar um arquivo de extensões rtf  
RTF: `$ doxygen -e rtf extensionsFile`

Quando `-s` eh especificado, os comentarios sao omitidos no arquivo de configuracao

Quando `configName` eh omitido, 'Doxyfile' sera utilizado por padrao



# Doxygen na Linha de Comando

```
$ doxygen --help
```

- 1) Use o doxygen para gerar um arquivo template de configuracao:  
\$ doxygen [-s] -g [configName]
- 2) Use o doxygen para atualizar um arquivo de configuracao antigo:  
\$ doxygen [-s] -u [configName]
- 3) Use o doxygen para gerar a documentacao atraves de um arquivo de configuracao existente:  
\$ doxygen [configName]
- 4) Use o doxygen para gerar um arquivo de estilos template para os formatos RTF, HTML e Latex.  
RTF: \$ doxygen -w rtf styleSheetFile  
HTML: \$ doxygen -w html headerFile footerFile styleSheetFile [configFile]  
LaTeX: \$ doxygen -w latex headerFile styleSheetFile [configFile]
- 5) Use o doxygen para gerar um arquivo de extensões rtf  
RTF: \$ doxygen -e rtf extensionsFile

Quando -s eh especificado, os comentarios sao omitidos no arquivo de configuracao

Quando configName eh omitido, 'Doxyfile' sera utilizado por padrao

# Exemplo de Arquivo de Configuração

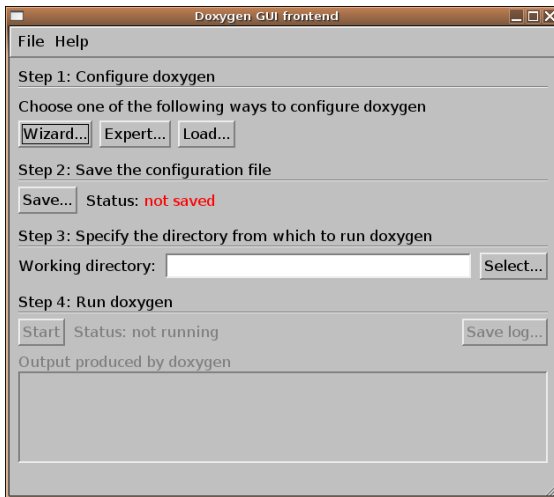
```
# Arquivo de configuracao Doxygen

PROJECT_NAME           = "LmaTimer : timer-cpp"
PROJECT_NUMBER         = 0.9
OUTPUT_DIRECTORY      = /home/axpavim/exemplos-treinamento/timer-cpp/doc/
OUTPUT_LANGUAGE        = English
FULL_PATH_NAMES        = YES
STRIP_FROM_PATH        = /home/axpavim/exemplos-treinamento/timer-cpp/
INPUT                  = /home/axpavim/exemplos-treinamento/timer-cpp
FILE_PATTERNS          = *.c \
                        *.cc \
                        *.cpp \
                        *.java \
                        *.h \
                        *.hpp \
                        *.cs \
                        *.py
GENERATE_HTML           = YES
HTML_OUTPUT            = html
HTML_FILE_EXTENSION    = .html
CLASS_DIAGRAMS         = YES
CLASS_GRAPH            = YES
COLLABORATION_GRAPH    = YES
GROUP_GRAPHS          = YES
.
.
.
```

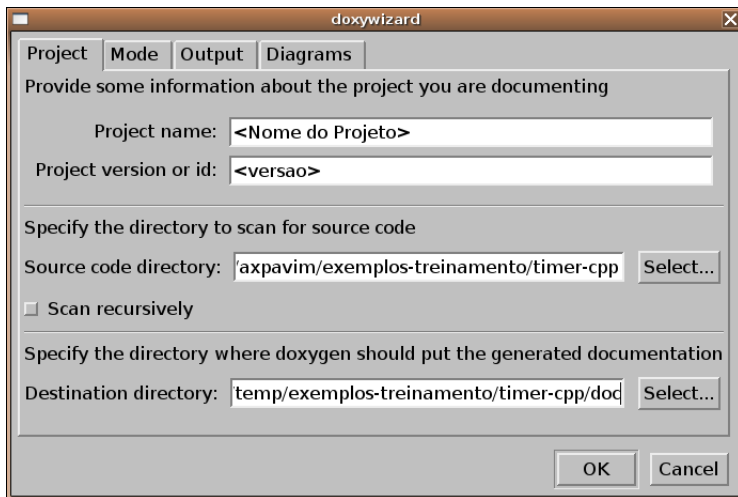
# Sumário da Subseção

- 1 Introdução
  - Importância da Documentação do Código-fonte
- 2 Documentando o Código-fonte com Doxygen
  - Estilos de Documentação
- 3 Utilização e Configuração de um Projeto Doxygen
  - A Ferramenta Doxygen
  - A Ferramenta Doxywizard
- 4 Resultado

# Interface Gráfica para utilização do Doxygen



# Configuração Básica (*Wizard*)

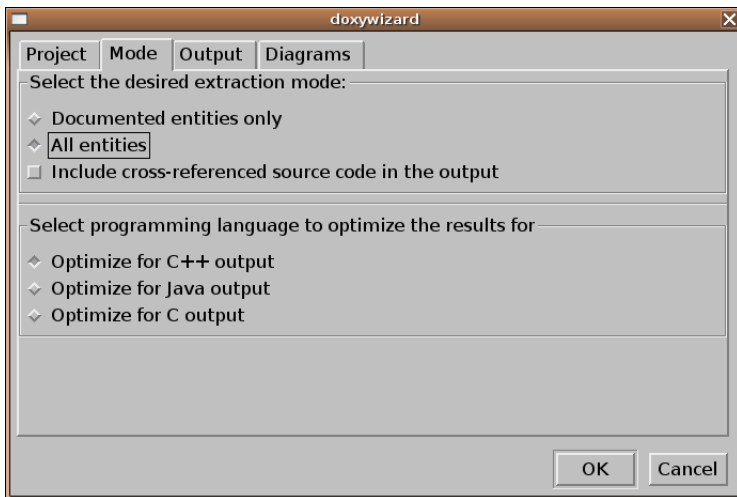


The screenshot shows the 'doxywizard' application window. It has a title bar with a close button (X) and a tabbed interface with four tabs: 'Project', 'Mode', 'Output', and 'Diagrams'. The 'Project' tab is selected. The main area contains three sections:

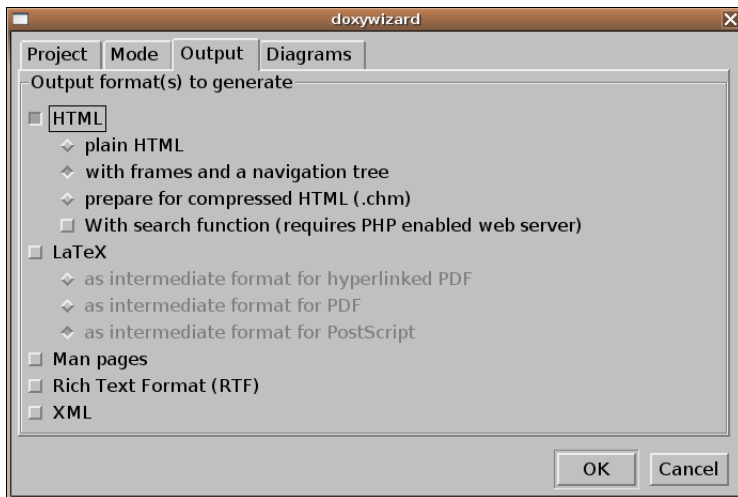
- Provide some information about the project you are documenting**
  - Project name: <Nome do Projeto>
  - Project version or id: <versao>
- Specify the directory to scan for source code**
  - Source code directory: 'axpavim/exemplos-treinamento/timer-cpp' (with a 'Select...' button)
  - ☐ Scan recursively
- Specify the directory where doxygen should put the generated documentation**
  - Destination directory: 'temp/exemplos-treinamento/timer-cpp/doc' (with a 'Select...' button)

At the bottom right, there are 'OK' and 'Cancel' buttons.

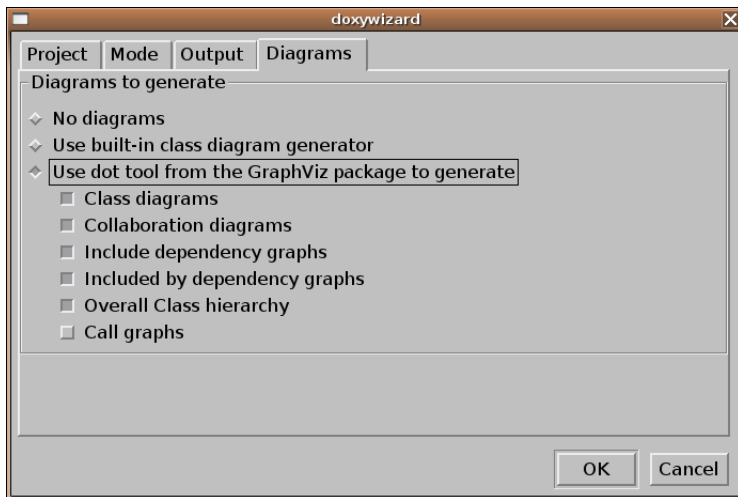
# Configuração Básica (*Wizard*)



# Configuração Básica (*Wizard*)

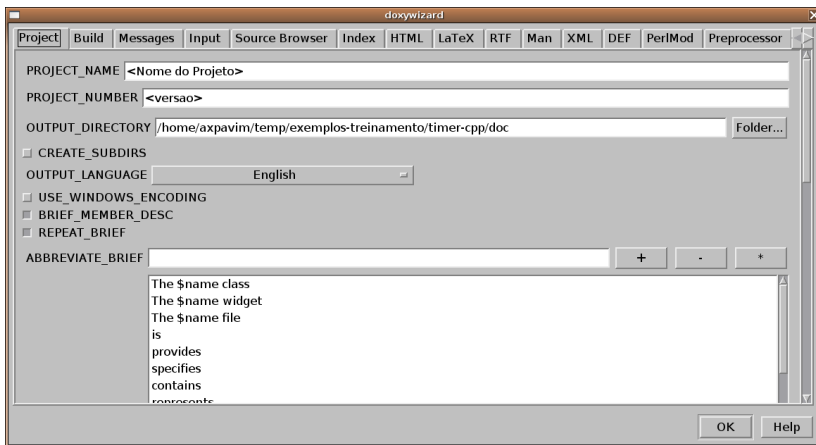


# Configuração Básica (*Wizard*)

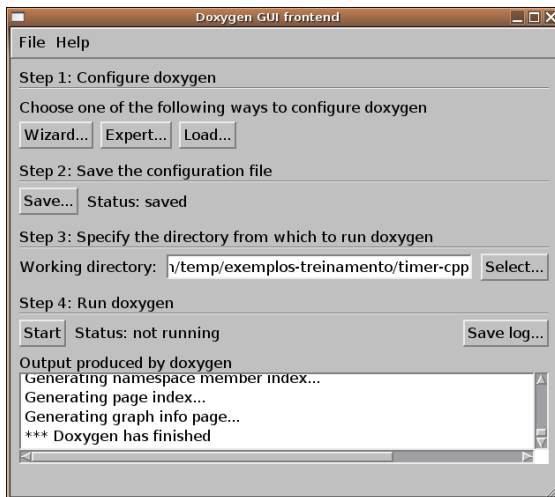




# Configuração Avançada (*Wizard*)



# Interface Gráfica para utilização do Doxygen



# Resultado final

The screenshot shows a web browser window titled "LmaTimer : timer-cpp - Mozilla Firefox". The address bar shows the file path: `file:///home/axpavim/exemplos-treinamento/timer-cpp/doc/html/index.html`. The browser has a menu bar with "Arquivo", "Editar", "Exibir", "Jr", "Favoritos", "Ferramentas", and "Ajuda". Below the menu bar is a toolbar with navigation icons and a search icon.

The main content area displays the "LmaTimer::LmaTimer Class Reference" page. At the top, there is a navigation bar with links: "Main Page", "Namespace List", "Class List", "File List", "Namespace Members", "Class Members", and "File Members". Below this, the page title "LmaTimer::LmaTimer" is shown. The main heading is "LmaTimer::LmaTimer Class Reference".

The text describes the class: "The LmaTimer class gives the user the functionalities of a timer. [More...](#)". It includes a code snippet: `#include <LmaTimer.h>` and a link to the "List of all members".

The "Public Member Functions" section lists the following:

- LmaTimer ()**  
Default Constructor.
- ~LmaTimer ()**  
Default Destructor.
- void SaveTimerStatisticsToFile (bool a\_bSaveStatistics)**  
Allow to choose if the creation/destruction statistics must be saved.
- string GetCurrentTime (bool a\_bDay=true, bool a\_bMonth=true, bool a\_bYear=true, bool a\_bHour=true, bool a\_bMinute=true, bool a\_bSecond=true, bool a\_bMillisecond=true)**  
Returns to the user the current system time, allowing some format configuration.
- void PrintCurrentTime (bool a\_bDay=true, bool a\_bMonth=true, bool a\_bYear=true, bool a\_bHour=true, bool a\_bMinute=true, bool a\_bSecond=true, bool a\_bMillisecond=true)**  
Prints the system time to the standard output.
- LMATIMERSTATUS Delay (int a\_iDelayValue=DEFAULT\_DELAY\_VALUE)**  
Performs a delay in the program execution.
- LMATIMERSTATUS StartChronometer ()**  
Starts chronometering an event.
- LMATIMERSTATUS StopChronometer ()**  
Stops chronometering an event.

On the left side, there is a sidebar with a tree view showing the project structure: "LmaTimer : timer-cpp" (expanded) with sub-items: "File List" (containing LmaTimer.cpp, LmaTimer.h, LmaTimerDef.h, LmaTimerDemo.cpp), "Class List" (containing LmaTimer::LmaTimer), "Class Members", "Namespace List" (containing LmaTimer), "std", "File Members", and "Namespace Members".

At the bottom left, there is a "Concluido" status indicator.

# Obrigado pela Atenção!

Alberto Xavier Pavim

`axpavim@das.ufsc.br`

`axp@labmetro.ufsc.br`

