

Example of a documented header file, which was converted using of **DocBuilder** to a documentation.

You find this example after the installation of **DocBuilder** in the subdirectory 'examples/cpp'.

If you use the files cpp.dbs and cpp.dbp unchanged for the creation of the documentation, then the documentation in the subdirectory 'examples/cpp/output/rtf' is created.

In the existing document you find the following contents:

Page 1: These introductory remarks
Page 2: Source code of the header file
Page 16: Generated documentation to the header file

```

/*****
    anadat.h          Klassen und Definitionen zur Datenverwaltung
                      sd, 3/98
*****/

#ifndef _anadat_h
#define _anadat_h

#include <set>
#include <stack>
#include <vector>
#include <ostream>

#include "cbl/cat.h"          // cMapObj

#include "dbb/ifil.h"
#include "dbb/parse.h"
#include "dbb/gstrlist.h"
#include "dbb/dbbhash.h"

#include "anaconst.h"
#include "anadefs.h"
#include "anace.h"
#include "anatree.h"
#include "anautil.h"        // gDocPos

//-----
//  @c  gCitation          Quellcode-Zitat
//-----

class gCitation : public cShared {
public:
    // @t  Status-Typ
    // Beschreibt, ob das Zitat vollstaendig, unvollstaendig (wegen EOF)
    // bzw. noch gar nicht vom File gelesen wurde.
    enum status_type { Complete, Incomplete, Missing };

    gCitation(const gSrcFile& f, int fst, int lst)
        : m_file(f), m_first(fst), m_last(lst), m_status(Missing)
    {
        cASSERT(1 <= m_first && m_first <= m_last + 1);
    }

    const gSrcFile& file() const          { return m_file; }
    int             first() const         { return m_first; }
    int             last() const          { return m_last; }

    // @m Der von ``get`` gelesene Zitattext.
    const gString&  text() const          { return m_text; }

    // @m Holt das Zitat aus dem File, falls `status() != Complete`.
    // ``status`` liefert das Ergebnis der Operation.
    void            get();

    // @m Der Status des Zitats (s. ``status_type``).
    status_type     status() const        { return m_status; }

private:
    gSrcFile        m_file;
    int             m_first;              // erste Zeile des Zitats
    int             m_last;              // letzte Zeile

```

```

    gString      m_text;          // der Zitattext
    status_type  m_status;

};

typedef cRef<gCitation> gCitRef;

//-----
// @c gPred          Prädikate
//
// @des
//   Prädikate sind Aussagen über Attribute, d.h. Mengen von Attributen
//   mit zugeordneten Wertelisten).
//   Die Werte können einschließlich ('ATTR == val1, val2, ...') oder
//   ausschließend ('ATTR != val1, val2, ...') angegeben werden.
//-----

class gPred {
public:
    gPred() {}

    gPred(const gString& attr, const gStrList& val_list, bool incl = true)
    {
        str_set val_set;
        gStrList::const_iterator it = val_list.begin(),
                                ie = val_list.end();
        for (; it != ie; ++it)
            val_set.insert(**it);
        m_attrs.insert(make_pair(attr, make_pair(val_set, incl)));
    }

    // @m Entscheidet, ob das Praedikat fuer die Attribut"liste" 'amap' gilt.
    bool holds(const str_map& amap) const;

    // @m Erweitert das Praedikat um die Attributliste von 'other'.
    // Diese ueberschreibt ggf. vorhandene Attribute.
    void add_predicate(const gPred& other);

    // @m Entfernt 'attr' aus dem Praedikat.
    // Es erfolgt keine Pruefung, ob 'attr' im Praedikat enthalten war.
    void remove_attr(const gString& attr) { m_attrs.erase(attr); }

private:
    typedef pair<str_set, bool> value_list;
    // Menge von Werten mit Inklusions-Flag

    map<gString, value_list> m_attrs;
    // zum Praedikat gehoerige Attribute
};

//-----
// @c gBookmark      Bookmark
//
// @des
//   Bookmarks haben einen Namen und eine dokumentweit eindeutige
//   Textmarke. Der Anwender arbeitet mit dem Namen, in der Dokumentation
//   erscheinen die Textmarken.
//-----

class gBookmark : public cShared {
public:

```

```

gBookmark(const gString& n = "")
    : m_name(n), m_txtlabel(make_label(n)) {}

// @m Erzeugt unter Verwendung des Namens 'n' eine eindeutige Textmarke
// mit gueltiger Syntax.
static gString make_label(const gString& n);

// @m Ist 'n' ein gueltiger Bookmark-Name?
static bool is_valid_name(const gString& n);

// @m Der Name der Bookmark.
const gString& name() const { return m_name; }

// @m Die Textmarke der Bookmark.
const gString& txtlabel() const { return m_txtlabel; }

private:

    enum { Max_len = 35 }; // max. Laenge von Bookmark-Idents

    static gCounter s_count; // fuer Numerierung der Textmarken

    gString m_name; // Name in der Eingabe
    gString m_txtlabel; // Textmarke fuer ATX-Ausgabe
};

typedef cRef<gBookmark> gBookRef;

//-----
// gEnvRef
//-----

struct gCmdEnv;

typedef cRef<gCmdEnv> gEnvRef;

//-----
// @c gDocNode Knoten im Dokumentationsbaum
//
// @des
// Dies ist die Basisklasse fuer alle Dokumentationseinheiten.
//-----

class gDocNode : public gNode<gDocNode> {
public:

    typedef gNode<gDocNode> super;

    gDocNode(const super* parent = 0)
        : super(parent) {}

    // @m Gibt die Unit und ihre Kinder aus (pre order).
    virtual void print(std::ostream& os);
};

typedef cRef<gDocNode> gDocRef;

//-----
// Rubrik-Konfiguration
//-----

```

```

namespace AnaDat {

    bool is_voiderr(int cfg);
    bool is_autofill(int cfg);
    bool is_verbatim(int cfg);

    bool is_voiderr(const gString& utype, const gString& subname);
    bool is_autofill(const gString& utype, const gString& subname);
    bool is_verbatim(const gString& utype, const gString& subname);

}

//-----
// @c gDocUnit          Dokumentationseinheit
//
// @des
//     Basisklasse fuer @cref<gSyntUnit>{Syntaxeinheiten} und
//     @cref<gTextUnit>{Texteinheiten}.
//-----

class gDocUnit;

typedef cRef<gDocUnit> gUnitRef;

class gDocUnit : public gDocNode {
public:

    enum flag_type {
        Autobook = 1, Autoxe = 2,
        Print = 4, Online = 8, Sequential = 16,
        Max = 8          // Dummy
    };

    gDocUnit(const gStrList& qn, const gSrcFile& f, int line)
        : m_id(s_serial++)
        , m_style("")
        , m_loc(f, line)
        , m_ce(0)
        , m_ce_prio(0)
        , m_flags(Autobook | Autoxe | Print | Online | Sequential)
    {
        qname(qn);          // setzt auch 'm_autobook'
    }

    virtual ~gDocUnit() {}

    // @m Erzeugt flache Kopie der Doku-Einheit (fuer mehrfache Ausgabe).
    // Die Auto-Bookmark der Kopie erhaelt eine neue Textmarke.
    virtual gUnitRef clone() const = 0;

    // Bestimmungsstuecke

    // @m Der unqualifizierte Name
    gString name() const;

    // @m Der qualifizierte Name als Liste der Komponenten
    const gStrList& qname() const { return m_qname; }
    void qname(const gStrList& qn)
    {
        m_qname = qn;
        m_autobook = gBookmark(qn.make_string());
    }
}

```

```

// @m Der qualifizierte Name als String
gString      qual_name() const
{
    return m_qname.make_string(src_loc().file().name_sep());
}

gString      class_name() const;
// die vorletzte Komponente des qualifizierten Namens
// (kann auch Namespace sein)

// Eigenschaften

unsigned      id() const          { return m_id; }

// @m Der Ausgabestil der Unit
const gString& style() const      { return m_style; }
void          style(const gString& s) { m_style = s; }

// @m Der Standort der Unit in der Dokumentation
const gLoc&   loc() const         { return m_loc; }

// @m Der Standort des zugeordneten Codeelements (falls vorhanden) bzw.
// der Unit (anderenfalls)
const gLoc&   src_loc() const
{
    return (m_ce ? m_ce->loc() : m_loc);
}

// @m Ordnet der Unit das Codeelement 'ce' mit der Prioritaet 'prio' zu.
void          set_ce(const gCERef& ce, int prio);

// @m Das zugeordnete Codeelement
const gCERef& ce() const          { return m_ce; }

// @m Welche Prioritaet hat das zugeordnete CE (0: kein CE zugeordnet)?
int          ce_prio() const      { return m_ce_prio; }

// @m Uebernimmt die Informationen aus dem CE.
void          get_ce_infos();

// @m Die Position im Dokumentationsbaum
const gDocPos& doc_pos() const    { return m_docpos; }
void          doc_pos(const gDocPos& pos) { m_docpos = pos; }

// @m Verschiedene Flags
int&         flags()              { return m_flags; }

// Attribute

// @m Der Wert von Attribut 'attr' fuer diese Unit
// @ret "undefined" falls unbelegt
gString      attr_value(const gString& attr) const
{
    str_map::const_iterator it = m_attrs.find(attr);
    return (it != m_attrs.end() ? it->second : "undefined");
}

// @m Die Attribut-Map der Unit
const str_map& attributes() const { return m_attrs; }

// @m Setzt Attribut 'attr' auf Wert 'val'.
void          set_attr(const gString& attr, const gString& val)
{
    m_attrs[attr] = val;
}

// Bookmarks und Index-Eintraege

```

```

// @m Das Textlabel der Auto-Bookmark
virtual gString      autobook() const      { return m_autobook.txtlabel(); }

// @m Weist den Bookmarks der Unit (Auto, explizite und XEs) neue
// Textmarken zu und aktualisiert die Eintraege in der Bookmark-Tabelle.
// (fuer ``clone``)
void                clone_bookmarks();

// Rubriken

// @m Liefert Inhalt der Rubrik 'sub'. Fuegt eine leere Rubrik dieses
// Namens ein, falls noch nicht existent.
const gStrList&     subentry(const gString& sub) const { return m_subs[sub]; }

// @m Fuegt der Rubrik 'sub' den Text 'text' hinzu.
void                add_subentry(const gString& sub, const gStrList& text);

// @m Ersetzt den Inhalt der Rubrik 'sub' durch 'text'.
void                set_subentry(const gString& sub, const gStrList& text);

// Linkline

// @m Die Verweiszeile der Unit
gStrList&           linkline()             { return m_linkline; }
const gStrList&     linkline() const      { return m_linkline; }

const gStrList&     srclink() const        { return m_srclink; }

// @m Nimmt den Source-Link mit Verweistext 'lnktxt' in die Linkline auf.
// @com 'lnktxt' darf nicht leer sein.
void                srclink(const gStrList& lnktxt)
{
    cASSERT(! lnktxt.empty());
    m_srclink = lnktxt;
}

// Doku-Check

// @m Ueberprueft die Unit auf Vollstaendigkeit und Zulaessigkeit
// der Rubriken.
virtual void        check_doc() const      {}

// ATX-Codierung

// @m Wertet die Umgebung 'env' aus (fuer Auto-Bookmark, Src-Link,
// Variablen).
// @com Wird beim Eintragen der Unit in den Dokumentationsbaum gerufen.
virtual void        env_eval(const gEnvRef& env);

// @m Gibt die Unit auf 'os' aus.
virtual void        print(std::ostream& os) { cASSERT(0); }

protected:

// @m Bestimmt den Ausgabe-Stil aus:
// -. ``style``
// -. dem 'UNITTYPE'-Wert
virtual gString     get_style() const;

// @m Fuehrt Substitutionen mit Umgebungsbezug (z.B. Umgebungsvariablen,
// Tabellen) in den Rubriktexten aus.
virtual void        env_substitute(const gEnvRef& env);

// @m Fuehrt Substitutionen mit lokalem oder globalem Bezug
// (z.B. Unit-Variablen, Ermitteln der Verweisziele) in den
// Rubriktexten aus.

```

```

virtual void      process_subentries();

virtual void      substitute(gStrList& txt, gStrList::iterator& it,
                             bool verbatim = false) const;
    // Fuehrt in 'txt' an der durch 'it' bezeichneten Position
    // ggf. notwendige Substitutionen aus (abhaengig von 'verbatim').

virtual void      print_header(std::ostream& os) const;
    // Gibt den Unit-Header aus

// @m Iteriert 'row' entsprechend 'env' (aktueller Filter u. '$curclass')
// bei Vorhandensein einer Unit-Variable oder eines '@outattr'-Kommandos.
void              iterate_row(gStrList& row, const gEnvRef& env) const;

mutable map<gString, gStrList>  m_subs;
    // die Rubriken
    // 'mutable' fuer 'subentry()' - Einfuegen leerer Rubriken ist
    // trotz 'const' mgl.

int              m_flags;

private:

    static unsigned    s_serial;        // lfd. Nummer

    gStrList           m_qname;         // der qualifizierte Name
    unsigned           m_id;           // global eindeutige ID-Nummer
    gString            m_style;        // Format-Stil
    gStrMap            m_attrs;        // Attribute
    gLoc               m_loc;         // Position im Quelltext
    gCERef             m_ce;          // zugeh. CE (fuer Srclink)
    int                m_ce_prio;     // die Prioritaet des zugeh. CE
                                // 0: kein CE

    gBookmark          m_autobook;     // die Standard-Bookmark
    gStrList           m_linkline;     // Linkline
    gStrList           m_srclink;      // Source-Link-Text
                                // leer: keinen Srclink erzeugen
    gDocPos            m_docpos;      // Position im Dokumentationsbaum
};

struct gLessUnitName : public binary_function<gUnitRef, gUnitRef, bool> {
    bool operator()(const gUnitRef& lhs, const gUnitRef& rhs) const
    {
        cASSERT(lhs && rhs);
        return lhs->qual_name() < rhs->qual_name();
    }
};

struct gLessUnitLoc : public binary_function<gUnitRef, gUnitRef, bool> {
    bool operator()(const gUnitRef& lhs, const gUnitRef& rhs) const
    {
        cASSERT(lhs && rhs);
        return lhs->loc() < rhs->loc();
    }
};

struct gLessUnitCELoc : public binary_function<gUnitRef, gUnitRef, bool> {
    bool operator()(const gUnitRef& lhs, const gUnitRef& rhs) const
    {
        cASSERT(lhs && rhs);
        return lhs->src_loc() < rhs->src_loc();
    }
};

```

```

    }
};

/* wegen der nachtraeglich erzeugten Kommentar-Units nicht sinnvoll
struct gLessUnitId : public binary_function<gUnitRef, gUnitRef, bool> {

    bool operator()(const gUnitRef& lhs, const gUnitRef& rhs) const
    {
        cASSERT(lhs && rhs);
        return (lhs->id() < rhs->id());
    }
};
*/

//-----
// @c gTextUnit          Text-Einheit
//
// @des
// Text-Einheiten enthalten Fliesstext, der im Unterschied zu den
// @cref<gSyntUnit>{Syntaxeinheiten} nicht in Rubriken gegliedert ist.
//-----

class gTextUnit : public gDocUnit {
public:

    gTextUnit(const gStrList& qn, const gSrcFile& f, int line)
        : gDocUnit(qn, f, line) {}

    // @m Erzeugt flache Kopie der Text-Einheit (fuer mehrfache Ausgabe).
    // Die Bookmarks des Resultats erhalten eine eigene Textmarke.
    virtual gUnitRef    clone() const;

    // @m Fuegt der Einheit den Text 'text' hinzu.
    void                add_text(const gStrList& text)
    {
        add_subentry("text", text);
    }

    // @m Liefert Inhalt der Einheit.
    const gStrList&     text() const          { return subentry("text"); }

    // @m Gibt die Einheit auf 'os' aus.
    virtual void        print(std::ostream& os);

protected:

    virtual void        print_header(std::ostream& os) const;
};

//-----
// @c gSyntUnit          Syntax-Einheit
//
// @des
// Syntaxeinheiten sind im Gegensatz zu den @cref<gTextUnit>{Texteinheiten}
// in @cref<subentry>{Rubriken} gegliedert. Diese können bei der Ausgabe
// durch eine @cref<tpl>{Ausgabe-Schablone} konfiguriert werden.
//-----

class gSyntUnit : public gDocUnit {
public:

```

```

gSyntUnit(const gStrList& qn, const gSrcFile& f, int line);

// @m Erzeugt flache Kopie der Doku-Einheit (fuer mehrfache Ausgabe).
// Die Bookmarks des Resultats erhalten eigene Textmarken.
virtual gUnitRef    clone() const;

// @m Die Ausgabe-Schablone der Unit
const gString&      tpl() const           { return m_tpl; }
void               tpl(const gString& s)
{
    cASSERT(! s.empty()); m_tpl = s;
}

// @m Die (einzige) Headline der Unit
const gStrList&    hdlne() const         { return m_hdlne; }
void               hdlne(const gStrList& hdl);

// @m Setzt Tiefe der auszugebenden Klassenhierarchie auf 'up' bzw. 'down'
// (fuer Klassen-Einheiten).
// @com
// - Verlangt: 'attr_value(UNITTYPE) == "class"'
// - Verlangt: 'up > 0 || up == gClassHierarchy::Full || up ==
gClassHierarchy::None'
// - dto. fuer 'down'
void               set_hierarchy(int up, int down);

// @m Ueberprueft die Unit auf Vollstaendigkeit und Zulaessigkeit der
// Rubriken.
virtual void        check_doc() const;

// @m Wertet die Umgebung 'env' aus (fuer Auto-Bookmark, Src-Link,
// Variablen).
// @com Wird beim Eintragen der Unit in den Dokumentationsbaum gerufen.
virtual void        env_eval(const gEnvRef& env);

// @m Gibt die Unit auf 'os' aus.
virtual void        print(std::ostream& os);

protected:

// @m Bestimmt den Ausgabe-Stil aus:
// -. ''style''
// -. Template-Style
// -. 'UNITTYPE'-Wert
virtual gString     get_style();

private:

gStrList           m_hdlne;           // die (einzige) Headline der Unit
gString            m_tpl;             // Unit-Template
int                m_hierup;         // Umfang der Klassenhierarchie
int                m_hierdown;       // nach oben bzw. unten
};

//-----
// @c gSectHeader      Ein ATX-Section-Header
//-----

class gSectHeader : public gDocNode {
public:

gSectHeader(const gString& outfn = "")
    : m_ofname(outfn), m_tpl("STPL_Default") {}

// @m Der Name des Section-Templates

```

```

const gString& tpl() const { return m_tpl; }
void          tpl(const gString& s)
{
    cASSERT(! s.empty()); m_tpl = s;
}

// @m Gibt den Header samt zugehoeriger Section auf die Datei 'm_ofname'
// aus oder auf 'os', wenn 'm_ofname' leer ist.
void          print(std::ostream& os);

private:

// @m Gibt Section-Header auf 'os' aus.
virtual void  print_header(std::ostream& os) const;

gString      m_ofname;      // Name der Ausgabedatei (voller Pfad)
gString      m_tpl;        // Name des Section-Templates
};

typedef cRef<gSectHeader> gSectRef;

//-----
// @c gFilter          Filter für @cref<gDocUnit>{Doku-Einheiten}
//
// @des
// Ein Filter beschreibt eine Menge von Doku-Einheiten durch
// - Angabe von gültigen Dateinamen und -pfaden
// - ein @cref<gPred>{Prädikat} über Attribute
// - ggf. einen Klassennamen (innerhalb '@b_outclass[filter]' ...)
//-----

class gFilter {
public:

// @t Filter-Wertetyp
// Beschreibt den Inhalt des Filters: ein Vektor von Unit-Referenzen
// ('vector' wegen Sortierbarkeit mit 'std::sort')
typedef std::vector<gUnitRef> value_type;

// @t Sortier-Typ
// Sortierung nach Name, Unit-Location bzw. CE-Location
enum sort_type { Sort_name, Sort_uloc, Sort_celoc };

gFilter()
    : m_curdir(gFileSystem::get_curdir()), m_miss_out(true)
    , m_sort(Sort_celoc) {}

gFilter(const gPred& p)
    : m_curdir(gFileSystem::get_curdir()), m_pred(p), m_miss_out(true)
    , m_sort(Sort_celoc) {}

// @m Der Wert (Inhalt) des Filters: die Menge aller Doku-Einheiten,
// die diesem Filter genuegen.
value_type value() const;

// @m Setzt den Filter auf den Ausgangszustand zurueck:
// alle Dokueinheiten im Filter, Sortierung in urspruengl. Reihenfolge
void reset();

// Filtern nach Dateien

// @m Setzt das aktuelle Verzeichnis fuer die File-Filterung auf 'pabs'.
// @com 'pabs' muss absolut sein.
// @ret Erfolg der Operation

```

```

bool        cur_dir(const gPathstring& pabs);

// @m Setzt den Pfadfilter auf 'plist'.
void        set_path_filter(const gPathList& plist)    { m_pflt = plist; }

// @m Setzt den Filefilter auf 'flist'.
void        set_file_filter(const gPathList& flist)   { m_fflt = flist; }

// Filtern nach Praedikaten

// @m Fuegt dem Filter das Praedikat 'pred' hinzu.
// @com Die einzelnen Praedikate werden UND-verknuepft.
void        add_predicate(const gPred& pred) { m_pred.add_predicate(pred); }

// @m Setzt 'pred' als alleiniges Filter-Praedikat.
void        set_predicate(const gPred& pred)    { m_pred = pred; }

// @m Setzt den Klassennamen 'cname'. Im Filter sind anschliessend nur noch
// Einheiten, deren vorletzte Namenskomponente gleich 'cname' ist.
void        set_classname(const gString& cname) { m_cname = cname; }

// @m Entfernt den Klassennamen aus dem Filter.
void        unset_classname()                  { m_cname = ""; }

// @m Entfernt das Attribut 'attr' aus dem Filterpraedikat.
// @com Es erfolgt keine Pruefung, ob 'attr' im Praedikat enthalten war.
void        remove_attr(const gString& attr)
{
    m_pred.remove_attr(attr);
}

// @m Setzt das Sortier-Flag.
// @par
// - 'true': Der Inhalt des Filters ('value') wird alphabetisch sortiert.
// - 'false': Es erfolgt keine Sortierung; die Units behalten ihre
//            urspruengliche Reihenfolge.
void        set_sort(sort_type s)              { m_sort = s; }

private:

    static bool    missing_ce(const gUnitRef u);

    bool          in_filter(const gUnitRef u) const;

    gPathstring   m_curdir;           // das aktuelle Verzeichnis
    gPathList     m_pflt;             // der Pfadfilter
    gPathList     m_fflt;             // der Filefilter
    gPred         m_pred;
    gString       m_cname;
    mutable bool  m_miss_out;         // Units ohne CE ausgeben?
    sort_type     m_sort;

};

//-----
// @c gDocTable      Tabelle von Doku-Einheiten
//-----

class gDocTable : public gHash<list<gUnitRef> > {
public:

    // @m Traegt die Unit 'u' sowohl unter ihrem vollstaendig qualifizierten
    // Namen als auch mit sukzessive dequalifiziertem Namen ein, um das
    // Aufloesen unqualifizierter Verweise zu ermoeeglichen.
    void        insert(const gUnitRef u);

};

```

```

//-----
// @c gDocManager      Aufbau und Verwaltung der Dokumentation
//
// @des
//     Diese Klasse bietet Methoden zur Strukturierung der Dokumentation
//     und zum Einfügen von Units.
//-----

class gDocManager : public gGlobalObj {

public:

    typedef gDocNode      tree_type;
    typedef tree_type::reference    reference;

    // @m Initialisierung beim Programmstart
    virtual void    init()                { m_uccount = 0; }

    // @m Reinitialisierung fuer neue Dokumentation
    virtual void    reset();

    // @m Die Anzahl der Units in der Dokumentation
    unsigned        unit_count() const    { return m_uccount; }

    // @m Liefert den aktuellen Headline-Level.
    unsigned        hdlevel() const      { return m_hdlevel; }

    // @m Setzt Headline-Level auf 'n'.
    // @com Verlangt: '1 <= n <= AnaConst::Hdl_max'
    void            set_hdlevel(unsigned n);

    // @m Eroeffnet neuen Abschnitt mit Header 'hdr' und setzt Hdl-Level auf 1.
    void            start_section(const gSectRef& hdr);

    // @m Fuegt die Unit 'u' am Ende der Kindliste des aktuellen Knotens ein.
    void            add_unit(const gUnitRef& u);

    // @m Fuegt die Klassen-Unit 'c' am Ende der Kindliste des aktuellen
    // Knotens ein und ordnet ihm die nachfolgenden Knoten unter.
    void            add_class(const gUnitRef& c);

    // @m Beendet einen mit ''add_class'' eingefuegten Klassenknoten.
    void            end_class();

    // @m Sucht die Bookmark 'bm' unter den expliziten (globalen) Bookmarks
    // und im Dokumentationsbaum (die von 'start' aus "dichteste") und
    // belegt 'lbl' mit ihrer Textmarke (falls gefunden).
    // @ret Erfolg der Suche
    bool            find_bookmark(const gDocPos& start, const gString& bm,
                                gString& lbl) const;

    // @m Gibt die aufgebraute Dokumentation auf 'os' aus.
    void            print(std::ostream& os)    { m_tree.print(os); }

    void            dump(std::ostream& os) const    { m_tree.dump(os); }
    // Debug-Ausgabe des Baumes auf 'os'

private:

    tree_type        m_tree;
    // der Baum der ausgegebenen Doku-Einheiten

    reference        m_active[AnaConst::Hdl_max];
    // die "aktiven" Knoten, denen die Units der entsprechenden
    // Hdl-Levels als Kinder zugeordnet werden
    // 'm_active[0]' ist die Baumwurzel, ihre Kinder sind die Sections.

```

```

reference      m_current;
    // der aktive Knoten des niedrigsten offenen Hdl-Levels

unsigned      m_hdlevel;
    // der aktuelle Hdl-Level

gDocTable     m_doctab;
    // die Knoten im Dokumentationsbaum (fuer Verweiszielsuche)

gDocPos       m_docpos;
    // die Position der zuletzt eingefuegten Doku-Einheit

unsigned      m_ucount;
    // Anzahl der ausgegebenen Units
};

//-----
// @c gCmdEnv      Kommando-Umgebung
//-----

struct gCmdEnv : public cShared {

    // @v  der aktuelle Filter
    gFilter      cur_filter;

    // @v  der Stack der aktuellen Klassen
    std::stack<gString> cur_class;

    // @v  das aktive Section-Template
    gString      cur_stpl;
};

//-----
//      gDBAttributes      Die DBS-Settings
//-----

class gDBAttributes : public gGlobalObj, public cMapObj {

public:

    virtual void    init();
        // Eintragen der vordefinierten Attribute

    virtual void    reset()          { datamap().clear(); init(); }
        // Reinitialisierung fuer neue Dokumentation
};

//-----
//      Globale Objekte
//-----

namespace AnaDat {

    extern gDocManager      doc;
        // der Verwalter der auszugebenden Doku-Einheiten

    extern gGlobalHash<int>      bkmk_table;
        // die expliziten Bookmarks
        // Wert 0: Bookmark wurde noch nicht ausgegeben
        // Wert 1: Bookmark wurde schon ausgegeben
        // (fuer 'clone_bookmarks')

    extern gGlobalHash<gCitRef>      cit_table;
        // die Quellcode-Zitate
};

```

```
extern gGlobalHash<int>          output_files;
    // die Namen der bereits erzeugten ATX-Files
    // (fuer Auswahl von write- bzw. append-Modus)

extern gDBAttributes             dbs;
    // die Manual-Definitionen
}

#endif // _anadat_h
```

Inhaltsverzeichnis

Klassen.....	20
gCitation.....	20
gCitation::status_type.....	20
gCitation::text.....	20
gCitation::get.....	20
gCitation::status.....	20
gPred.....	20
gPred::holds.....	21
gPred::add_predicate.....	21
gPred::remove_attr.....	21
gBookmark.....	21
gBookmark::make_label.....	21
gBookmark::is_valid_name.....	21
gBookmark::name.....	21
gBookmark::txtlabel.....	22
gDocNode.....	22
gDocNode::print.....	22
gDocUnit.....	22
gDocUnit::clone.....	22
gDocUnit::name.....	23
gDocUnit::qname.....	23
gDocUnit::qual_name.....	23
gDocUnit::style.....	23
gDocUnit::loc.....	23
gDocUnit::src_loc.....	23
gDocUnit::set_ce.....	23
gDocUnit::ce.....	23
gDocUnit::ce_prio.....	23
gDocUnit::get_ce_infos.....	23
gDocUnit::doc_pos.....	24
gDocUnit::flags.....	24
gDocUnit::attr_value.....	24
gDocUnit::attributes.....	24
gDocUnit::set_attr.....	24
gDocUnit::autobook.....	24
gDocUnit::clone_bookmarks.....	24
gDocUnit::subentry.....	24
gDocUnit::add_subentry.....	24
gDocUnit::set_subentry.....	25
gDocUnit::linkline.....	25
gDocUnit::srclink.....	25
gDocUnit::check_doc.....	25
gDocUnit::env_eval.....	25
gDocUnit::print.....	25
gDocUnit::get_style.....	25
gDocUnit::env_substitute.....	25
gDocUnit::process_subentries.....	25
gDocUnit::iterate_row.....	26
gTextUnit.....	26
gTextUnit::clone.....	26
gTextUnit::add_text.....	26
gTextUnit::text.....	26
gTextUnit::print.....	26
gSyntUnit.....	26

gSyntUnit::clone	27
gSyntUnit::tpl.....	27
gSyntUnit::hdline	27
gSyntUnit::set_hierarchy	27
gSyntUnit::check_doc	27
gSyntUnit::env_eval.....	27
gSyntUnit::print	28
gSyntUnit::get_style.....	28
gSectHeader	28
gSectHeader::tpl.....	28
gSectHeader::print.....	28
gSectHeader::print_header	28
gFilter	28
gFilter::value_type	29
gFilter::sort_type.....	29
gFilter::value.....	29
gFilter::reset	29
gFilter::cur_dir	29
gFilter::set_path_filter	29
gFilter::set_file_filter.....	29
gFilter::add_predicate.....	30
gFilter::set_predicate	30
gFilter::set_classname	30
gFilter::unset_classname.....	30
gFilter::remove_attr.....	30
gFilter::set_sort	30
gDocTable	30
gDocTable::insert.....	31
gDocManager	31
gDocManager::init	31
gDocManager::reset	31
gDocManager::unit_count	31
gDocManager::hdlevel	31
gDocManager::set_hdlevel	31
gDocManager::start_section	32
gDocManager::add_unit	32
gDocManager::add_class.....	32
gDocManager::end_class.....	32
gDocManager::find_bookmark.....	32
gDocManager::print	32
gCmdEnv	32
gCmdEnv::cur_filter.....	32
gCmdEnv::cur_class.....	33
gCmdEnv::cur_stpl	33
Typdefinitionen	34
gCitation::status_type.....	34
gFilter::value_type	34
gFilter::sort_type.....	34
Die Quickreferenz	35
gBookmark	35
gBookmark::is_valid_name.....	35
gBookmark::make_label.....	35
gBookmark::name.....	35
gBookmark::txtlabel.....	35
gCitation	35
gCitation::get	35

gCitation::status	35
gCitation::status_type.....	36
gCitation::text	36
gCmdEnv	36
gCmdEnv::cur_class	36
gCmdEnv::cur_filter.....	36
gCmdEnv::cur_stpl	36
gDocManager	36
gDocManager::add_class	36
gDocManager::add_unit.....	36
gDocManager::end_class	37
gDocManager::find_bookmark.....	37
gDocManager::hdlevel	37
gDocManager::init	37
gDocManager::print	37
gDocManager::reset	37
gDocManager::set_hdlevel.....	37
gDocManager::start_section	37
gDocManager::unit_count.....	37
gDocNode.....	38
gDocNode::print	38
gDocTable	38
gDocTable::insert.....	38
gDocUnit	38
gDocUnit::add_subentry.....	38
gDocUnit::attr_value.....	38
gDocUnit::attributes.....	38
gDocUnit::autobook.....	38
gDocUnit::ce.....	39
gDocUnit::ce_prio.....	39
gDocUnit::check_doc.....	39
gDocUnit::clone.....	39
gDocUnit::clone_bookmarks	39
gDocUnit::doc_pos	39
gDocUnit::env_eval	39
gDocUnit::env_substitute	39
gDocUnit::flags.....	39
gDocUnit::get_ce_infos.....	40
gDocUnit::get_style	40
gDocUnit::iterate_row.....	40
gDocUnit::linkline.....	40
gDocUnit::loc.....	40
gDocUnit::name.....	40
gDocUnit::print	40
gDocUnit::process_subentries	40
gDocUnit::qname	40
gDocUnit::qual_name	41
gDocUnit::set_attr.....	41
gDocUnit::set_ce.....	41
gDocUnit::set_subentry.....	41
gDocUnit::src_loc	41
gDocUnit::srclink.....	41
gDocUnit::style.....	41
gDocUnit::subentry	41
gFilter	41
gFilter::add_predicate.....	42

gFilter::cur_dir	42
gFilter::remove_attr.....	42
gFilter::reset	42
gFilter::set_classname	42
gFilter::set_file_filter.....	42
gFilter::set_path_filter	42
gFilter::set_predicate	42
gFilter::set_sort	42
gFilter::sort_type.....	43
gFilter::unset_classname.....	43
gFilter::value.....	43
gFilter::value_type	43
gPred.....	43
gPred::add_predicate.....	43
gPred::holds	43
gPred::remove_attr.....	43
gSectHeader	43
gSectHeader::print.....	44
gSectHeader::print_header	44
gSectHeader::tpl.....	44
gSyntUnit.....	44
gSyntUnit::check_doc	44
gSyntUnit::clone	44
gSyntUnit::env_eval.....	44
gSyntUnit::get_style.....	44
gSyntUnit::hdline	45
gSyntUnit::print	45
gSyntUnit::set_hierarchy	45
gSyntUnit::tpl.....	45
gTextUnit.....	45
gTextUnit::add_text	45
gTextUnit::clone	45
gTextUnit::print	45
gTextUnit::text.....	45
Index.....	46

Klassen

gCitation (Klasse)

Quellcode-Zitat

Klassenhierarchie

cShared (*public*)

gCitation

Deklaration class gCitation : public cShared

Basisklassen public cShared

gCitation::status_type (Typ)

Status-Typ

Beschreibung Beschreibt, ob das Zitat vollstaendig, unvollstaendig (wegen EOF) bzw. noch gar nicht vom File gelesen wurde.

Deklaration enum status_type { Complete, Incomplete, Missing };

gCitation::text (Methode)

Beschreibung Der von *get* [S. 20] gelesene Zitattext.

Deklaration const gString &text() const;

gCitation::get (Methode)

Beschreibung Holt das Zitat aus dem File, falls status() != Complete. *status* [S. 20] liefert das Ergebnis der Operation.

Deklaration void get();

gCitation::status (Methode)

Beschreibung Der Status des Zitats (s. *status_type* [S. 20]).

Deklaration status_type status() const;

gPred (Klasse)

Prädikate

Klassenhierarchie

gPred

Beschreibung Prädikate sind Aussagen über Attribute, d.h. Mengen von Attributen mit zugeordneten Wertelisten). Die Werte können einschließend (ATTR == val1, val2, ...) oder ausschließend (ATTR != val1, val2, ...) angegeben werden.

Deklaration class gPred

Basisklassen

gPred::holds **(Methode)**

Beschreibung Entscheidet, ob das Praedikat fuer die Attribut"liste" amap gilt.

Deklaration `bool holds(const str_map& amap) const;`

gPred::add_predicate **(Methode)**

Beschreibung Erweitert das Praedikat um die Attributliste von other. Diese ueberschreibt ggf. vorhandene Attribute.

Deklaration `void add_predicate(const gPred& other);`

gPred::remove_attr **(Methode)**

Beschreibung Entfernt attr aus dem Praedikat. Es erfolgt keine Pruefung, ob attr im Praedikat enthalten war.

Deklaration `void remove_attr(const gString& attr);`

gBookmark **(Klasse)**

Bookmark

Klassenhierarchie

cShared (*public*)

gBookmark

Beschreibung Bookmarks haben einen Namen und eine dokumentweit eindeutige Textmarke. Der Anwender arbeitet mit dem Namen, in der Dokumentation erscheinen die Textmarken.

Deklaration `class gBookmark : public cShared`

Basisklassen `public cShared`

gBookmark::make_label **(Methode)**

Beschreibung Erzeugt unter Verwendung des Namens n eine eindeutige Textmarke mit gueltiger Syntax.

Deklaration `static gString make_label(const gString& n);`

gBookmark::is_valid_name **(Methode)**

Beschreibung Ist n ein gueltiger Bookmark-Name?

Deklaration `static bool is_valid_name(const gString& n);`

gBookmark::name **(Methode)**

Beschreibung Der Name der Bookmark.

Deklaration `const gString &name() const;`

gBookmark::txtlabel **(Methode)**

Beschreibung Die Textmarke der Bookmark.

Deklaration `const gString &txtlabel() const;`

gDocNode **(Klasse)**

Knoten im Dokumentationsbaum**Klassenhierarchie**

gNode<gDocNode> (*public*)

gDocNode

gDocUnit (*public*)

gTextUnit (*public*)

gSyntUnit (*public*)

gSectHeader (*public*)

Beschreibung Dies ist die Basisklasse fuer alle Dokumentationseinheiten.

Deklaration `class gDocNode : public gNode<gDocNode>`

Basisklassen `public gNode<gDocNode>`

gDocNode::print **(Methode)**

Beschreibung Gibt die Unit und ihre Kinder aus (pre order).

Deklaration `virtual void print(std::ostream& os);`

gDocUnit **(Klasse)**

Dokumentationseinheit**Klassenhierarchie**

gNode<gDocNode> (*public*)

gDocNode (*public*)

gDocUnit

gTextUnit (*public*)

gSyntUnit (*public*)

Beschreibung Basisklasse fuer *Syntaxeinheiten* [S. 26] und *Texteinheiten* [S. 26].

Deklaration `class gDocUnit : public gDocNode`

Basisklassen `public gDocNode`

gDocUnit::clone **(Methode)**

Beschreibung Erzeugt flache Kopie der Doku-Einheit (fuer mehrfache Ausgabe). Die Auto-Bookmark der Kopie erhaelt eine neue Textmarke.

Deklaration `virtual gUnitRef clone() const = 0;`

gDocUnit::name *(Methode)*

Beschreibung Der unqualifizierte Name

Deklaration `gString name() const;`

gDocUnit::qname *(Methode)*

Beschreibung Der qualifizierte Name als Liste der Komponenten

Deklaration `const gStrList &qname() const;`

gDocUnit::qual_name *(Methode)*

Beschreibung Der qualifizierte Name als String

Deklaration `gString qual_name() const;`

gDocUnit::style *(Methode)*

Beschreibung Der Ausgabestil der Unit

Deklaration `const gString &style() const;`

gDocUnit::loc *(Methode)*

Beschreibung Der Standort der Unit in der Dokumentation

Deklaration `const gLoc &loc() const;`

gDocUnit::src_loc *(Methode)*

Beschreibung Der Standort des zugeordneten Codeelements (falls vorhanden) bzw. der Unit (anderenfalls)

Deklaration `const gLoc &src_loc() const;`

gDocUnit::set_ce *(Methode)*

Beschreibung Ordnet der Unit das Codeelement `ce` mit der Prioritaet `prio` zu.

Deklaration `void set_ce(const gCERef& ce, int prio);`

gDocUnit::ce *(Methode)*

Beschreibung Das zugeordnete Codeelement

Deklaration `const gCERef &ce() const;`

gDocUnit::ce_prio *(Methode)*

Beschreibung Welche Prioritaet hat das zugeordnete CE (0: kein CE zugeordnet)?

Deklaration `int ce_prio() const;`

gDocUnit::get_ce_infos *(Methode)*

Beschreibung Uebernimmt die Informationen aus dem CE.

Deklaration `void get_ce_infos();`

gDocUnit::doc_pos *(Methode)*

Beschreibung Die Position im Dokumentationsbaum

Deklaration `const gDocPos &doc_pos() const;`

gDocUnit::flags *(Methode)*

Beschreibung Verschiedene Flags

Deklaration `int &flags();`

gDocUnit::attr_value *(Methode)*

Beschreibung Der Wert von Attribut `attr` fuer diese Unit

Deklaration `gString attr_value(const gString& attr) const;`

Rückgabe "undefined" falls unbelegt

gDocUnit::attributes *(Methode)*

Beschreibung Die Attribut-Map der Unit

Deklaration `const str_map &attributes() const;`

gDocUnit::set_attr *(Methode)*

Beschreibung Setzt Attribut `attr` auf Wert `val`.

Deklaration `void set_attr(const gString& attr, const gString& val);`

gDocUnit::autobook *(Methode)*

Beschreibung Das Textlabel der Auto-Bookmark

Deklaration `virtual gString autobook() const;`

gDocUnit::clone_bookmarks *(Methode)*

Beschreibung Weist den Bookmarks der Unit (Auto, explizite und XEs) neue Textmarken zu und aktualisiert die Eintraege in der Bookmark-Tabelle. (fuer *clone* [S. 22])

Deklaration `void clone_bookmarks();`

gDocUnit::subentry *(Methode)*

Beschreibung Liefert Inhalt der Rubrik `sub`. Fuegt eine leere Rubrik dieses Namens ein, falls noch nicht existent.

Deklaration `const gStrList &subentry(const gString& sub) const;`

gDocUnit::add_subentry *(Methode)*

Beschreibung Fuegt der Rubrik `sub` den Text `text` hinzu.

Deklaration `void add_subentry(const gString& sub, const gStrList& text);`

gDocUnit::set_subentry **(Methode)**

Beschreibung Ersetzt den Inhalt der Rubrik `sub` durch `text`.

Deklaration `void set_subentry(const gString& sub, const gStrList& text);`

gDocUnit::linkline **(Methode)**

Beschreibung Die Verweiszeile der Unit

Deklaration `gStrList &linkline();`

gDocUnit::srclink **(Methode)**

Beschreibung Nimmt den Source-Link mit Verweistext `lnktxt` in die Linkline auf.

Deklaration `void srclink(const gStrList& lnktxt);`

Anmerkung `lnktxt` darf nicht leer sein.

gDocUnit::check_doc **(Methode)**

Beschreibung Ueberprueft die Unit auf Vollstaendigkeit und Zulaessigkeit der Rubriken.

Deklaration `virtual void check_doc() const;`

gDocUnit::env_eval **(Methode)**

Beschreibung Wertet die Umgebung `env` aus (fuer Auto-Bookmark, Src-Link, Variablen).

Deklaration `virtual void env_eval(const gEnvRef& env);`

Anmerkung Wird beim Eintragen der Unit in den Dokumentatonsbaum gerufen.

gDocUnit::print **(Methode)**

Beschreibung Gibt die Unit auf `os` aus.

Deklaration `virtual void print(std::ostream& os);`

gDocUnit::get_style **(Methode)**

Beschreibung Bestimmt den Ausgabe-Stil aus: -. *style* [S. 23] -. dem UNITTYPE-Wert

Deklaration `virtual gString get_style() const;`

gDocUnit::env_substitute **(Methode)**

Beschreibung Fuehrt Substitutionen mit Umgebungsbezug (z.B. Umgebungsvariablen, Tabellen) in den Rubriktexten aus.

Deklaration `virtual void env_substitute(const gEnvRef& env);`

gDocUnit::process_subentries **(Methode)**

Beschreibung Fuehrt Substitutionen mit lokalem oder globalem Bezug (z.B. Unit-Variablen,

Ermitteln der Verweisziele) in den Rubriktexten aus.

Deklaration `virtual void process_subentries();`

gDocUnit::iterate_row *(Methode)*

Beschreibung Iteriert row entsprechend env (aktueller Filter u. gDocUnit) bei Vorhandensein einer Unit-Variable oder eines @outattr-Kommandos.

Deklaration `void iterate_row(gStrList& row, const gEnvRef& env) const;`

gTextUnit *(Klasse)*

Text-Einheit

Klassenhierarchie

<code>gNode<gDocNode></code> (<i>public</i>)
<code>gDocNode</code> (<i>public</i>)
<code>gDocUnit</code> (<i>public</i>)
<code>gTextUnit</code>

Beschreibung Text-Einheiten enthalten Fliesstext, der im Unterschied zu den *Syntaxeinheiten* [S. 26] nicht in Rubriken gegliedert ist.

Deklaration `class gTextUnit : public gDocUnit`

Basisklassen `public gDocUnit`

gTextUnit::clone *(Methode)*

Beschreibung Erzeugt flache Kopie der Text-Einheit (fuer mehrfache Ausgabe). Die Bookmarks des Resultats erhalten eine eigene Textmarke.

Deklaration `virtual gUnitRef clone() const;`

gTextUnit::add_text *(Methode)*

Beschreibung Fuegt der Einheit den Text text hinzu.

Deklaration `void add_text(const gStrList& text);`

gTextUnit::text *(Methode)*

Beschreibung Liefert Inhalt der Einheit.

Deklaration `const gStrList &text() const;`

gTextUnit::print *(Methode)*

Beschreibung Gibt die Einheit auf os aus.

Deklaration `virtual void print(std::ostream& os);`

gSyntUnit *(Klasse)*

Syntax-Einheit

Klassenhierarchie

gNode<gDocNode> (*public*)

gDocNode (*public*)

gDocUnit (*public*)

gSyntUnit

Beschreibung Syntaxeinheiten sind im Gegensatz zu den *Texteinheiten* [S. 26] in *Rubriken* [S. 24] gegliedert. Diese können bei der Ausgabe durch eine *Ausgabe-Schablone* [S. 27] konfiguriert werden.

Deklaration `class gSyntUnit : public gDocUnit`

Basisklassen `public gDocUnit`

gSyntUnit::clone *(Methode)*

Beschreibung Erzeugt flache Kopie der Doku-Einheit (fuer mehrfache Ausgabe). Die Bookmarks des Resultats erhalten eigene Textmarken.

Deklaration `virtual gUnitRef clone() const;`

gSyntUnit::tpl *(Methode)*

Beschreibung Die Ausgabe-Schablone der Unit

Deklaration `const gString &tpl() const;`

gSyntUnit::hdline *(Methode)*

Beschreibung Die (einzige) Headline der Unit

Deklaration `const gStrList &hdline() const;`

gSyntUnit::set_hierarchy *(Methode)*

Beschreibung Setzt Tiefe der auszugebenden Klassenhierarchie auf up bzw. down (fuer Klassen-Einheiten).

Deklaration `void set_hierarchy(int up, int down);`

Anmerkung

- Verlangt: `attr_value(UNITTYPE) == "class"` - Verlangt: `up > 0 || up == gClassHierarchy::Full || up == gClassHierarchy::None` - dto. fuer down

gSyntUnit::check_doc *(Methode)*

Beschreibung Ueberprueft die Unit auf Vollstaendigkeit und Zulaessigkeit der Rubriken.

Deklaration `virtual void check_doc() const;`

gSyntUnit::env_eval *(Methode)*

Beschreibung Wertet die Umgebung env aus (fuer Auto-Bookmark, Src-Link, Variablen).

Deklaration `virtual void env_eval(const gEnvRef& env);`

Anmerkung Wird beim Eintragen der Unit in den Dokumentatonsbaum gerufen.

gSyntUnit::print **(Methode)**

Beschreibung Gibt die Unit auf os aus.

Deklaration `virtual void print (std::ostream& os);`

gSyntUnit::get_style **(Methode)**

Beschreibung Bestimmt den Ausgabe-Stil aus: -. *style* [S. 23] -. Template-Style -. UNITTYPE-Wert

Deklaration `virtual gString get_style();`

gSectHeader **(Klasse)**

Ein ATX-Section-Header

Klassenhierarchie

<code>gNode<gDocNode> (public)</code>
<code>gDocNode (public)</code>
gSectHeader

Deklaration `class gSectHeader : public gDocNode`

Basisklassen `public gDocNode`

gSectHeader::tpl **(Methode)**

Beschreibung Der Name des Section-Templates

Deklaration `const gString &tpl() const;`

gSectHeader::print **(Methode)**

Beschreibung Gibt den Header samt zugehoeriger Section auf die Datei m_ofname aus oder auf os, wenn m_ofname leer ist.

Deklaration `void print (std::ostream& os);`

gSectHeader::print_header **(Methode)**

Beschreibung Gibt Section-Header auf os aus.

Deklaration `virtual void print_header (std::ostream& os) const;`

gFilter **(Klasse)**

Filter für Doku-Einheiten [S. 22]

Klassenhierarchie

gFilter

Beschreibung Ein Filter beschreibt eine Menge von Doku-Einheiten durch - Angabe von gültigen Dateinamen und -pfaden - ein *Prädikat* [S. 20] über Attribute - ggf. einen Klassennamen (innerhalb \@b_outclass[filter] ...)

Deklaration `class gFilter`

Basisklassen

gFilter::value_type **(Typ)**

Filter-Wertetyp

Beschreibung Beschreibt den Inhalt des Filters: ein Vektor von Unit-Referenzen (vector wegen Sortierbarkeit mit `std::sort`)

Deklaration `typedef std::vector<gUnitRef> value_type;`

gFilter::sort_type **(Typ)**

Sortier-Typ

Beschreibung Sortierung nach Name, Unit-Location bzw. CE-Location

Deklaration `enum sort_type { Sort_name, Sort_uloc, Sort_celoc };`

gFilter::value **(Methode)**

Beschreibung Der Wert (Inhalt) des Filters: die Menge aller Doku-Einheiten, die diesem Filter genuegen.

Deklaration `value_type value() const;`

gFilter::reset **(Methode)**

Beschreibung Setzt den Filter auf den Ausgangszustand zurueck: alle Dokueinheiten im Filter, Sortierung in urspruengl. Reihenfolge

Deklaration `void reset();`

gFilter::cur_dir **(Methode)**

Beschreibung Setzt das aktuelle Verzeichnis fuer die File-Filterung auf pabs.

Deklaration `bool cur_dir(const gPathstring& pabs);`

Rückgabe Erfolg der Operation

Anmerkung pabs muss absolut sein.

gFilter::set_path_filter **(Methode)**

Beschreibung Setzt den Pfadfilter auf plist.

Deklaration `void set_path_filter(const gPathList& plist);`

gFilter::set_file_filter **(Methode)**

Beschreibung Setzt den Filefilter auf flist.

Deklaration `void set_file_filter(const gPathList& flist);`

gFilter::add_predicate *(Methode)*

Beschreibung Fuegt dem Filter das Praedikat `pred` hinzu.
Deklaration `void add_predicate(const gPred& pred);`
Anmerkung Die einzelnen Praedikate werden UND-verknuepft.

gFilter::set_predicate *(Methode)*

Beschreibung Setzt `pred` als alleiniges Filter-Praedikat.
Deklaration `void set_predicate(const gPred& pred);`

gFilter::set_classname *(Methode)*

Beschreibung Setzt den Klassennamen `cname`. Im Filter sind anschliessend nur noch Einheiten, deren vorletzte Namenskomponente gleich `cname` ist.
Deklaration `void set_classname(const gString& cname);`

gFilter::unset_classname *(Methode)*

Beschreibung Entfernt den Klassennamen aus dem Filter.
Deklaration `void unset_classname();`

gFilter::remove_attr *(Methode)*

Beschreibung Entfernt das Attribut `attr` aus dem Filterpraedikat.
Deklaration `void remove_attr(const gString& attr);`
Anmerkung Es erfolgt keine Pruefung, ob `attr` im Praedikat enthalten war.

gFilter::set_sort *(Methode)*

Beschreibung Setzt das Sortier-Flag.
Deklaration `void set_sort(sort_type s);`
Parameter

- `true`: Der Inhalt des Filters (*value [S. 29]*) wird alphabetisch sortiert. - `false`: Es erfolgt keine Sortierung; die Units behalten ihre urspruengliche Reihenfolge.

gDocTable *(Klasse)*

Tabelle von Doku-Einheiten

Klassenhierarchie

`gHash<list<gUnitRef> >` (*public*)

gDocTable

Deklaration `class gDocTable : public gHash<list<gUnitRef> >`

Basisklassen `public gHash<list<gUnitRef> >`

gDocTable::insert **(Methode)**

Beschreibung Traegt die Unit u sowohl unter ihrem vollstaendig qualifizierten Namen als auch mit sukzessive dequalifiziertem Namen ein, um das Auflösen unqualifizierter Verweise zu ermöglichen.

Deklaration `void insert(const gUnitRef u);`

gDocManager **(Klasse)**

Aufbau und Verwaltung der Dokumentation**Klassenhierarchie**

gGlobalObj (<i>public</i>)

gDocManager

Beschreibung Diese Klasse bietet Methoden zur Strukturierung der Dokumentation und zum Einfügen von Units.

Deklaration `class gDocManager : public gGlobalObj`

Basisklassen `public gGlobalObj`

gDocManager::init **(Methode)**

Beschreibung Initialisierung beim Programmstart

Deklaration `virtual void init();`

gDocManager::reset **(Methode)**

Beschreibung Reinitialisierung fuer neue Dokumentation

Deklaration `virtual void reset();`

gDocManager::unit_count **(Methode)**

Beschreibung Die Anzahl der Units in der Dokumentation

Deklaration `unsigned unit_count() const;`

gDocManager::hdlevel **(Methode)**

Beschreibung Liefert den aktuellen Headline-Level.

Deklaration `unsigned hdlevel() const;`

gDocManager::set_hdlevel **(Methode)**

Beschreibung Setzt Headline-Level auf n.

Deklaration `void set_hdlevel(unsigned n);`

Anmerkung Verlangt: $1 \leq n \leq \text{AnaConst::Hdl_max}$

gDocManager::start_section **(Methode)**

Beschreibung Eröffnet neuen Abschnitt mit Header `hdr` und setzt Hdl-Level auf 1.

Deklaration `void start_section(const gSectRef& hdr);`

gDocManager::add_unit **(Methode)**

Beschreibung Fügt die Unit `u` am Ende der Kindliste des aktuellen Knotens ein.

Deklaration `void add_unit(const gUnitRef& u);`

gDocManager::add_class **(Methode)**

Beschreibung Fügt die Klassen-Unit `c` am Ende der Kindliste des aktuellen Knotens ein und ordnet ihm die nachfolgenden Knoten unter.

Deklaration `void add_class(const gUnitRef& c);`

gDocManager::end_class **(Methode)**

Beschreibung Beendet einen mit *add_class* [S. 32] eingefügten Klassenknoten.

Deklaration `void end_class();`

gDocManager::find_bookmark **(Methode)**

Beschreibung Sucht die Bookmark `bm` unter den expliziten (globalen) Bookmarks und im Dokumentationsbaum (die von `start` aus "dichteste") und belegt `lbl` mit ihrer Textmarke (falls gefunden).

Deklaration `bool find_bookmark(const gDocPos& start, const gString& bm, gString& lbl) const;`

Rückgabe Erfolg der Suche

gDocManager::print **(Methode)**

Beschreibung Gibt die aufgebraute Dokumentation auf `os` aus.

Deklaration `void print(std::ostream& os);`

gCmdEnv **(Klasse)**

Kommando-Umgebung

Klassenhierarchie

`cShared` (*public*)

`gCmdEnv`

Deklaration `struct gCmdEnv : public cShared`

Basisklassen `public cShared`

gCmdEnv::cur_filter **(Member)**

der aktuelle Filter

Deklaration `gFilter cur_filter;`

gCmdEnv::cur_class **(Member)**

der Stack der aktuellen Klassen

Deklaration `std::stack<gString> cur_class;`

gCmdEnv::cur_stpl **(Member)**

das aktive Section-Template

Deklaration `gString cur_stpl;`

Typdefinitionen

gCitation::status_type

(Typ)

Status-Typ

Beschreibung Beschreibt, ob das Zitat vollstaendig, unvollstaendig (wegen EOF) bzw. noch gar nicht vom File gelesen wurde.

Deklaration `enum status_type { Complete, Incomplete, Missing };`

gFilter::value_type

(Typ)

Filter-Wertetyp

Beschreibung Beschreibt den Inhalt des Filters: ein Vektor von Unit-Referenzen (vector wegen Sortierbarkeit mit `std::sort`)

Deklaration `typedef std::vector<gUnitRef> value_type;`

gFilter::sort_type

(Typ)

Sortier-Typ

Beschreibung Sortierung nach Name, Unit-Location bzw. CE-Location

Deklaration `enum sort_type { Sort_name, Sort_uloc, Sort_celoc };`

Die Quickreferenz

gBookmark [S. 21]

Bookmark

Beschreibung Bookmarks haben einen Namen und eine dokumentweit eindeutige Textmarke. Der Anwender arbeitet mit dem Namen, in der Dokumentation erscheinen die Textmarken.

Deklaration `class gBookmark : public cShared`

gBookmark::is_valid_name [S. 21]

Beschreibung Ist *n* ein gueltiger Bookmark-Name?

Deklaration `static bool is_valid_name(const gString& n);`

gBookmark::make_label [S. 21]

Beschreibung Erzeugt unter Verwendung des Namens *n* eine eindeutige Textmarke mit gueltiger Syntax.

Deklaration `static gString make_label(const gString& n);`

gBookmark::name [S. 21]

Beschreibung Der Name der Bookmark.

Deklaration `const gString &name() const;`

gBookmark::txlabel [S. 22]

Beschreibung Die Textmarke der Bookmark.

Deklaration `const gString &txlabel() const;`

gCitation [S. 20]

Quellcode-Zitat

Deklaration `class gCitation : public cShared`

gCitation::get [S. 20]

Beschreibung Holt das Zitat aus dem File, falls `status() != Complete`. *status* [S. 20] liefert das Ergebnis der Operation.

Deklaration `void get();`

gCitation::status [S. 20]

Beschreibung Der Status des Zitats (s. *status_type* [S. 20]).

Deklaration `status_type status() const;`

gCitation::status_type [S. 20]

Status-Typ

Deklaration `enum status_type { Complete, Incomplete, Missing };`

gCitation::text [S. 20]

Beschreibung Der von *get [S. 20]* gelesene Zitattext.

Deklaration `const gString &text() const;`

gCmdEnv [S. 32]

Kommando-Umgebung

Deklaration `struct gCmdEnv : public cShared`

gCmdEnv::cur_class [S. 33]

der Stack der aktuellen Klassen

Deklaration `std::stack<gString> cur_class;`

gCmdEnv::cur_filter [S. 32]

der aktuelle Filter

Deklaration `gFilter cur_filter;`

gCmdEnv::cur_stpl [S. 33]

das aktive Section-Template

Deklaration `gString cur_stpl;`

gDocManager [S. 31]

Aufbau und Verwaltung der Dokumentation

Beschreibung Diese Klasse bietet Methoden zur Strukturierung der Dokumentation und zum Einfügen von Units.

Deklaration `class gDocManager : public gGlobalObj`

gDocManager::add_class [S. 32]

Beschreibung Fuegt die Klassen-Unit *c* am Ende der Kindliste des aktuellen Knotens ein und ordnet ihm die nachfolgenden Knoten unter.

Deklaration `void add_class(const gUnitRef& c);`

gDocManager::add_unit [S. 32]

Beschreibung Fuegt die Unit *u* am Ende der Kindliste des aktuellen Knotens ein.

Deklaration `void add_unit(const gUnitRef& u);`

gDocManager::end_class [S. 32]

Beschreibung Beendet einen mit *add_class [S. 32]* eingefuegten Klassenknoten.

Deklaration `void end_class();`

gDocManager::find_bookmark [S. 32]

Beschreibung Sucht die Bookmark `bm` unter den expliziten (globalen) Bookmarks und im Dokumentationsbaum (die von `start` aus "dichteste") und belegt `lbl` mit ihrer Textmarke (falls gefunden).

Deklaration `bool find_bookmark(const gDocPos& start, const gString& bm, gString& lbl) const;`

gDocManager::hdlevel [S. 31]

Beschreibung Liefert den aktuellen Headline-Level.

Deklaration `unsigned hdlevel() const;`

gDocManager::init [S. 31]

Beschreibung Initialisierung beim Programmstart

Deklaration `virtual void init();`

gDocManager::print [S. 32]

Beschreibung Gibt die aufgebaute Dokumentation auf `os` aus.

Deklaration `void print(std::ostream& os);`

gDocManager::reset [S. 31]

Beschreibung Reinitialisierung fuer neue Dokumentation

Deklaration `virtual void reset();`

gDocManager::set_hdlevel [S. 31]

Beschreibung Setzt Headline-Level auf `n`.

Deklaration `void set_hdlevel(unsigned n);`

gDocManager::start_section [S. 32]

Beschreibung Eroeffnet neuen Abschnitt mit Header `hdr` und setzt HdL-Level auf 1.

Deklaration `void start_section(const gSectRef& hdr);`

gDocManager::unit_count [S. 31]

Beschreibung Die Anzahl der Units in der Dokumentation

Deklaration `unsigned unit_count() const;`

gDocNode [S. 22]

Knoten im Dokumentationsbaum

Beschreibung Dies ist die Basisklasse fuer alle Dokumentationseinheiten.

Deklaration `class gDocNode : public gNode<gDocNode>`

gDocNode::print [S. 22]

Beschreibung Gibt die Unit und ihre Kinder aus (pre order).

Deklaration `virtual void print (std::ostream& os);`

gDocTable [S. 30]

Tabelle von Doku-Einheiten

Deklaration `class gDocTable : public gHash<list<gUnitRef> >`

gDocTable::insert [S. 31]

Beschreibung Traegt die Unit u sowohl unter ihrem vollstaendig qualifizierten Namen als auch mit sukzessive dequalifiziertem Namen ein, um das Aufloesen unqualifizierter Verweise zu ermoeeglichen.

Deklaration `void insert (const gUnitRef u);`

gDocUnit [S. 22]

Dokumentationseinheit

Beschreibung Basisklasse fuer *Syntaxeinheiten* [S. 26] und *Texteinheiten* [S. 26].

Deklaration `class gDocUnit : public gDocNode`

gDocUnit::add_subentry [S. 24]

Beschreibung Fuegt der Rubrik sub den Text text hinzu.

Deklaration `void add_subentry (const gString& sub, const gStrList& text);`

gDocUnit::attr_value [S. 24]

Beschreibung Der Wert von Attribut attr fuer diese Unit

Deklaration `gString attr_value (const gString& attr) const;`

gDocUnit::attributes [S. 24]

Beschreibung Die Attribut-Map der Unit

Deklaration `const str_map &attributes () const;`

gDocUnit::autobook [S. 24]

Beschreibung Das Textlabel der Auto-Bookmark

Deklaration `virtual gString autobook () const;`

gDocUnit::ce [S. 23]

Beschreibung Das zugeordnete Codeelement
Deklaration `const gCERef &ce() const;`

gDocUnit::ce_prio [S. 23]

Beschreibung Welche Prioritaet hat das zugeordnete CE (0: kein CE zugeordnet)?
Deklaration `int ce_prio() const;`

gDocUnit::check_doc [S. 25]

Beschreibung Ueberprueft die Unit auf Vollstaendigkeit und Zulaessigkeit der Rubriken.
Deklaration `virtual void check_doc() const;`

gDocUnit::clone [S. 22]

Beschreibung Erzeugt flache Kopie der Doku-Einheit (fuer mehrfache Ausgabe). Die Auto-Bookmark der Kopie erhaelt eine neue Textmarke.
Deklaration `virtual gUnitRef clone() const = 0;`

gDocUnit::clone_bookmarks [S. 24]

Beschreibung Weist den Bookmarks der Unit (Auto, explizite und XEs) neue Textmarken zu und aktualisiert die Eintraege in der Bookmark-Tabelle. (fuer *clone* [S. 22])
Deklaration `void clone_bookmarks();`

gDocUnit::doc_pos [S. 24]

Beschreibung Die Position im Dokumentationsbaum
Deklaration `const gDocPos &doc_pos() const;`

gDocUnit::env_eval [S. 25]

Beschreibung Wertet die Umgebung env aus (fuer Auto-Bookmark, Src-Link, Variablen).
Deklaration `virtual void env_eval(const gEnvRef& env);`

gDocUnit::env_substitute [S. 25]

Beschreibung Fuehrt Substitutionen mit Umgebungsbezug (z.B. Umgebungsvariablen, Tabellen) in den Rubriktexten aus.
Deklaration `virtual void env_substitute(const gEnvRef& env);`

gDocUnit::flags [S. 24]

Beschreibung Verschiedene Flags
Deklaration `int &flags();`

gDocUnit::get_ce_infos [S. 23]

Beschreibung Uebernimmt die Informationen aus dem CE.

Deklaration `void get_ce_infos();`

gDocUnit::get_style [S. 25]

Beschreibung Bestimmt den Ausgabe-Stil aus: -. *style [S. 23]* -. dem UNITTYPE-Wert

Deklaration `virtual gString get_style() const;`

gDocUnit::iterate_row [S. 26]

Beschreibung Iteriert row entsprechend env (aktueller Filter u.) bei Vorhandensein einer Unit-Variable oder eines @outattr-Kommandos.

Deklaration `void iterate_row(gStrList& row, const gEnvRef& env) const;`

gDocUnit::linkline [S. 25]

Beschreibung Die Verweiszeile der Unit

Deklaration `gStrList &linkline();`

gDocUnit::loc [S. 23]

Beschreibung Der Standort der Unit in der Dokumentation

Deklaration `const gLoc &loc() const;`

gDocUnit::name [S. 23]

Beschreibung Der unqualifizierte Name

Deklaration `gString name() const;`

gDocUnit::print [S. 25]

Beschreibung Gibt die Unit auf os aus.

Deklaration `virtual void print(std::ostream& os);`

gDocUnit::process_subentries [S. 25]

Beschreibung Fuehrt Substitutionen mit lokalem oder globalem Bezug (z.B. Unit-Variablen, Ermitteln der Verweisziele) in den Rubriktexten aus.

Deklaration `virtual void process_subentries();`

gDocUnit::qname [S. 23]

Beschreibung Der qualifizierte Name als Liste der Komponenten

Deklaration `const gStrList &qname() const;`

gDocUnit::qual_name [S. 23]

Beschreibung Der qualifizierte Name als String

Deklaration `gString qual_name() const;`

gDocUnit::set_attr [S. 24]

Beschreibung Setzt Attribut `attr` auf Wert `val`.

Deklaration `void set_attr(const gString& attr, const gString& val);`

gDocUnit::set_ce [S. 23]

Beschreibung Ordnet der Unit das Codeelement `ce` mit der Priorität `prio` zu.

Deklaration `void set_ce(const gCERef& ce, int prio);`

gDocUnit::set_subentry [S. 25]

Beschreibung Ersetzt den Inhalt der Rubrik `sub` durch `text`.

Deklaration `void set_subentry(const gString& sub, const gStrList& text);`

gDocUnit::src_loc [S. 23]

Beschreibung Der Standort des zugeordneten Codeelements (falls vorhanden) bzw. der Unit (anderenfalls)

Deklaration `const gLoc &src_loc() const;`

gDocUnit::srclink [S. 25]

Beschreibung Nimmt den Source-Link mit Verweistext `lnktxt` in die Linkline auf.

Deklaration `void srclink(const gStrList& lnktxt);`

gDocUnit::style [S. 23]

Beschreibung Der Ausgabestil der Unit

Deklaration `const gString &style() const;`

gDocUnit::subentry [S. 24]

Beschreibung Liefert Inhalt der Rubrik `sub`. Fügt eine leere Rubrik dieses Namens ein, falls noch nicht existent.

Deklaration `const gStrList &subentry(const gString& sub) const;`

gFilter [S. 28]

Filter für Doku-Einheiten [S. 22]

Beschreibung Ein Filter beschreibt eine Menge von Doku-Einheiten durch - Angabe von gültigen Dateinamen und -pfaden - ein *Prädikat* [S. 20] über Attribute - ggf. einen Klassennamen (innerhalb `\@b_outclass[filter]` ...)

Deklaration `class gFilter`

gFilter::add_predicate [S. 30]

Beschreibung Fuegt dem Filter das Praedikat `pred` hinzu.

Deklaration `void add_predicate(const gPred& pred);`

gFilter::cur_dir [S. 29]

Beschreibung Setzt das aktuelle Verzeichnis fuer die File-Filterung auf `pabs`.

Deklaration `bool cur_dir(const gPathstring& pabs);`

gFilter::remove_attr [S. 30]

Beschreibung Entfernt das Attribut `attr` aus dem Filterpraedikat.

Deklaration `void remove_attr(const gString& attr);`

gFilter::reset [S. 29]

Beschreibung Setzt den Filter auf den Ausgangszustand zurueck: alle Dokueinheiten im Filter, Sortierung in urspruengl. Reihenfolge

Deklaration `void reset();`

gFilter::set_classname [S. 30]

Beschreibung Setzt den Klassennamen `cname`. Im Filter sind anschliessend nur noch Einheiten, deren vorletzte Namenskomponente gleich `cname` ist.

Deklaration `void set_classname(const gString& cname);`

gFilter::set_file_filter [S. 29]

Beschreibung Setzt den Filefilter auf `flist`.

Deklaration `void set_file_filter(const gPathList& flist);`

gFilter::set_path_filter [S. 29]

Beschreibung Setzt den Pfadfilter auf `plist`.

Deklaration `void set_path_filter(const gPathList& plist);`

gFilter::set_predicate [S. 30]

Beschreibung Setzt `pred` als alleiniges Filter-Praedikat.

Deklaration `void set_predicate(const gPred& pred);`

gFilter::set_sort [S. 30]

Beschreibung Setzt das Sortier-Flag.

Deklaration `void set_sort(sort_type s);`

gFilter::sort_type [S. 29]

Sortier-Typ

Deklaration `enum sort_type { Sort_name, Sort_uloc, Sort_celoc };`

gFilter::unset_classname [S. 30]

Beschreibung Entfernt den Klassennamen aus dem Filter.

Deklaration `void unset_classname();`

gFilter::value [S. 29]

Beschreibung Der Wert (Inhalt) des Filters: die Menge aller Doku-Einheiten, die diesem Filter genuegen.

Deklaration `value_type value() const;`

gFilter::value_type [S. 29]

Filter-Wertetyp

Deklaration `typedef std::vector<gUnitRef> value_type;`

gPred [S. 20]

Prädikate

Beschreibung Prädikate sind Aussagen über Attribute, d.h. Mengen von Attributen mit zugeordneten Wertelisten). Die Werte können einschließend (`ATTR == val1, val2, ...`) oder ausschließend (`ATTR != val1, val2, ...`) angegeben werden.

Deklaration `class gPred`

gPred::add_predicate [S. 21]

Beschreibung Erweitert das Praedikat um die Attributliste von `other`. Diese ueberschreibt ggf. vorhandene Attribute.

Deklaration `void add_predicate(const gPred& other);`

gPred::holds [S. 21]

Beschreibung Entscheidet, ob das Praedikat fuer die Attribut"liste" `amap` gilt.

Deklaration `bool holds(const str_map& amap) const;`

gPred::remove_attr [S. 21]

Beschreibung Entfernt `attr` aus dem Praedikat. Es erfolgt keine Pruefung, ob `attr` im Praedikat enthalten war.

Deklaration `void remove_attr(const gString& attr);`

gSectHeader [S. 28]

Ein ATX-Section-Header

Deklaration `class gSectHeader : public gDocNode`

gSectHeader::print [S. 28]

Beschreibung Gibt den Header samt zugehoeriger Section auf die Datei `m_ofname` aus oder auf `os`, wenn `m_ofname` leer ist.

Deklaration `void print(std::ostream& os);`

gSectHeader::print_header [S. 28]

Beschreibung Gibt Section-Header auf `os` aus.

Deklaration `virtual void print_header(std::ostream& os) const;`

gSectHeader::tpl [S. 28]

Beschreibung Der Name des Section-Templates

Deklaration `const gString &tpl() const;`

gSyntUnit [S. 26]

Syntax-Einheit

Beschreibung Syntaxeinheiten sind im Gegensatz zu den *Texteinheiten [S. 26]* in *Rubriken [S. 24]* gegliedert. Diese können bei der Ausgabe durch eine *Ausgabe-Schablone [S. 27]* konfiguriert werden.

Deklaration `class gSyntUnit : public gDocUnit`

gSyntUnit::check_doc [S. 27]

Beschreibung Ueberprueft die Unit auf Vollstaendigkeit und Zulaessigkeit der Rubriken.

Deklaration `virtual void check_doc() const;`

gSyntUnit::clone [S. 27]

Beschreibung Erzeugt flache Kopie der Doku-Einheit (fuer mehrfache Ausgabe). Die Bookmarks des Resultats erhalten eigene Textmarken.

Deklaration `virtual gUnitRef clone() const;`

gSyntUnit::env_eval [S. 27]

Beschreibung Wertet die Umgebung `env` aus (fuer Auto-Bookmark, Src-Link, Variablen).

Deklaration `virtual void env_eval(const gEnvRef& env);`

gSyntUnit::get_style [S. 28]

Beschreibung Bestimmt den Ausgabe-Stil aus: `- style [S. 23]` `-` Template-Style `-` UNITTYPE-Wert

Deklaration `virtual gString get_style();`

gSyntUnit::hdline [S. 27]

Beschreibung Die (einzige) Headline der Unit
Deklaration `const gStrList &hdline() const;`

gSyntUnit::print [S. 28]

Beschreibung Gibt die Unit auf `os` aus.
Deklaration `virtual void print(std::ostream& os);`

gSyntUnit::set_hierarchy [S. 27]

Beschreibung Setzt Tiefe der auszugebenden Klassenhierarchie auf `up` bzw. `down` (fuer Klassen-Einheiten).
Deklaration `void set_hierarchy(int up, int down);`

gSyntUnit::tpl [S. 27]

Beschreibung Die Ausgabe-Schablone der Unit
Deklaration `const gString &tpl() const;`

gTextUnit [S. 26]

Text-Einheit

Beschreibung Text-Einheiten enthalten Fliesstext, der im Unterschied zu den *Syntaxeinheiten* [S. 26] nicht in Rubriken gegliedert ist.
Deklaration `class gTextUnit : public gDocUnit`

gTextUnit::add_text [S. 26]

Beschreibung Fuegt der Einheit den Text `text` hinzu.
Deklaration `void add_text(const gStrList& text);`

gTextUnit::clone [S. 26]

Beschreibung Erzeugt flache Kopie der Text-Einheit (fuer mehrfache Ausgabe). Die Bookmarks des Resultats erhalten eine eigene Textmarke.
Deklaration `virtual gUnitRef clone() const;`

gTextUnit::print [S. 26]

Beschreibung Gibt die Einheit auf `os` aus.
Deklaration `virtual void print(std::ostream& os);`

gTextUnit::text [S. 26]

Beschreibung Liefert Inhalt der Einheit.
Deklaration `const gStrList &text() const;`

Index

A

add_class 32, 36
add_predicate 21, 29, 42, 43
add_subentry 24, 38
add_text 26, 45
add_unit 32, 36
attr_value 24, 38
attributes 24, 38
autobook 24, 38

C

ce 23, 39
ce_prio 23, 39
check_doc 25, 27, 39, 44
clone 22, 26, 27, 39, 44, 45
clone_bookmarks 24, 39
cur_class 32, 36
cur_dir 29, 42
cur_filter 32, 36
cur_stpl 33, 36

D

doc_pos 24, 39

E

end_class 32, 37
env_eval 25, 27, 39, 44
env_substitute 25, 39

F

find_bookmark 32, 37
flags 24, 39

G

gBookmark 21
gBookmark::is_valid_name 21, 35
gBookmark::make_label 21, 35
gBookmark::name 21, 35
gBookmark::txtlabel 22, 35
gCitation 20
gCitation::get 20, 35
gCitation::status 20, 35
gCitation::status_type 20, 34, 36
gCitation::text 20, 36
gCmdEnv 32
gCmdEnv::cur_class 32, 36
gCmdEnv::cur_filter 32, 36
gCmdEnv::cur_stpl 33, 36
gDocManager 31
gDocManager::add_class 32, 36
gDocManager::add_unit 32, 36
gDocManager::end_class 32, 37
gDocManager::find_bookmark 32, 37
gDocManager::hdlevel 31, 37
gDocManager::init 31, 37

gDocManager::print 32, 37
gDocManager::reset 31, 37
gDocManager::set_hdlevel 31, 37
gDocManager::start_section 31, 37
gDocManager::unit_count 31, 37
gDocNode 22
gDocNode::print 22, 38
gDocTable 30
gDocTable::insert 30, 38
gDocUnit 22
gDocUnit::add_subentry 24, 38
gDocUnit::attr_value 24, 38
gDocUnit::attributes 24, 38
gDocUnit::autobook 24, 38
gDocUnit::ce 23, 39
gDocUnit::ce_prio 23, 39
gDocUnit::check_doc 25, 39
gDocUnit::clone 22, 39
gDocUnit::clone_bookmarks 24, 39
gDocUnit::doc_pos 24, 39
gDocUnit::env_eval 25, 39
gDocUnit::env_substitute 25, 39
gDocUnit::flags 24, 39
gDocUnit::get_ce_infos 23, 40
gDocUnit::get_style 25, 40
gDocUnit::iterate_row 26, 40
gDocUnit::linkline 25, 40
gDocUnit::loc 23, 40
gDocUnit::name 23, 40
gDocUnit::print 25, 40
gDocUnit::process_subentries 25, 40
gDocUnit::qname 23, 40
gDocUnit::qual_name 23, 41
gDocUnit::set_attr 24, 41
gDocUnit::set_ce 23, 41
gDocUnit::set_subentry 25, 41
gDocUnit::src_loc 23, 41
gDocUnit::srlink 25, 41
gDocUnit::style 23, 41
gDocUnit::subentry 24, 41
get 20, 35
get_ce_infos 23, 40
get_style 25, 28, 40, 44
gFilter 28
gFilter::add_predicate 29, 42
gFilter::cur_dir 29, 42
gFilter::remove_attr 30, 42
gFilter::reset 29, 42
gFilter::set_classname 30, 42
gFilter::set_file_filter 29, 42
gFilter::set_path_filter 29, 42
gFilter::set_predicate 30, 42
gFilter::set_sort 30, 42

gFilter::sort_type..... 29, 34, 43
 gFilter::unset_classname 30, 43
 gFilter::value..... 29, 43
 gFilter::value_type 29, 34, 43
 gPred 20
 gPred::add_predicate..... 21, 43
 gPred::holds 21, 43
 gPred::remove_attr..... 21, 43
 gSectHeader..... 28
 gSectHeader::print 28, 44
 gSectHeader::print_header 28, 44
 gSectHeader::tpl..... 28, 44
 gSyntUnit..... 26
 gSyntUnit::check_doc 27, 44
 gSyntUnit::clone 27, 44
 gSyntUnit::env_eval..... 27, 44
 gSyntUnit::get_style..... 28, 44
 gSyntUnit::hdline 27, 45
 gSyntUnit::print 28, 45
 gSyntUnit::set_hierarchy..... 27, 45
 gSyntUnit::tpl 27, 45
 gTextUnit..... 26
 gTextUnit::add_text 26, 45
 gTextUnit::clone 26, 45
 gTextUnit::print 26, 45
 gTextUnit::text..... 26, 45
H
 hdlevel 31, 37
 hdline..... 27, 45
 holds 21, 43
I
 init 31, 37
 insert..... 30, 38
 is_valid_name 21, 35
 iterate_row 26, 40
L
 linkline..... 25, 40
 loc..... 23, 40
M
 make_label..... 21, 35
N
 name 21, 23, 35, 40

P
 print..... 22, 25, 26, 28, 32, 37, 38, 40, 44, 45
 print_header 28, 44
 process_subentries 25, 40
Q
 qname..... 23, 40
 qual_name 23, 41
R
 remove_attr 21, 30, 42, 43
 reset..... 29, 31, 37, 42
S
 set_attr 24, 41
 set_ce 23, 41
 set_classname 30, 42
 set_file_filter 29, 42
 set_hdlevel 31, 37
 set_hierarchy 27, 45
 set_path_filter..... 29, 42
 set_predicate..... 30, 42
 set_sort..... 30, 42
 set_subentry 25, 41
 sort_type..... 29, 34, 43
 src_loc..... 23, 41
 srclink 25, 41
 start_section 31, 37
 status 20, 35
 status_type..... 20, 34, 36
 style..... 23, 41
 subentry..... 24, 41
T
 text 20, 26, 36, 45
 tpl..... 27, 28, 44, 45
 txtlabel 22, 35
U
 unit_count 31, 37
 unset_classname 30, 43
V
 value..... 29, 43
 value_type..... 29, 34, 43