Proyecto Integrador - 2do Semestre

Generado por Doxygen 1.8.4

Viernes, 31 de Mayo de 2013 19:56:28

Índice general

Capítulo 1

Indice jerárquico

1.1. Jerarquía de la clase

Esta lista de herencias esta ordenada aproximadamente por orden alfabético:

Cuestionario 7
Encriptador 9
Pregunta 11
  Pregunta5a10 13
  PreguntaOpcMult 15
Validador 17

Capítulo 2

Índice de clases

2.1. Lista de clases

Lista de las clases, estructuras, uniones e interfaces con una breve descripción:

Cuestionario 7
Encriptador 9
Pregunta 11
Pregunta5a10 13
PreguntaOpcMult 15
Validador 17

Capítulo 3

Indice de archivos

3.1. Lista de archivos

Lista de todos los archivos con descripciones breves:

ProyectoIntegrador2/**Cuestionario.cpp** . . . 21
ProyectoIntegrador2/**Cuestionario.h** . . . 21
ProyectoIntegrador2/**Encriptador.cpp** . . . 22
ProyectoIntegrador2/**Encriptador.h** . . . 24
ProyectoIntegrador2/**Preguntas.cpp** . . . 26
ProyectoIntegrador2/**Preguntas.h** . . . 27
ProyectoIntegrador2/**ProyectoIntegrador2.cpp** . . . 28
ProyectoIntegrador2/**stdafx.cpp** . . . **??**
ProyectoIntegrador2/**stdafx.h** . . . **??**
ProyectoIntegrador2/**targetver.h** . . . **??**

Capítulo 4

Documentación de las clases

4.1. Referencia de la Clase Cuestionario

`#include <Cuestionario.h>`

Métodos públicos

- **Cuestionario** ()
- ∼**Cuestionario** ()
- bool **GuardarCuestionario** ()
- void **hacerPreguntas** ()
- void **addPregunta** (**Pregunta** ∗)

4.1.1. Descripción detallada

Definición en la línea 6 del archivo Cuestionario.h.

4.1.2. Documentación del constructor y destructor

4.1.2.1. Cuestionario::Cuestionario ( )

Definición en la línea 12 del archivo Cuestionario.cpp.

```
13 {
14
15 }
```

4.1.2.2. Cuestionario::∼Cuestionario ( )

Definición en la línea 17 del archivo Cuestionario.cpp.

```
18 {
19
20 }
```

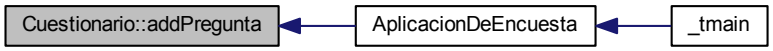
4.1.3. Documentación de las funciones miembro

4.1.3.1. void Cuestionario::addPregunta (Pregunta * *input*)

Definición en la línea 64 del archivo Cuestionario.cpp.

```
64                                          {
65     preguntas.push_back(input);
66
67 }
```

Gráfico de llamadas a esta función:



4.1.3.2. bool Cuestionario::GuardarCuestionario ( )

Definición en la línea 22 del archivo Cuestionario.cpp.

```
23 {
24     typedef std::chrono::system_clock Clock;
25     auto now = Clock::now();
26     std::time_t now_c = Clock::to_time_t(now);
27
28     struct tm *parts = std::localtime(&now_c);
29     int anio= 1900 + parts->tm_year;
30     int month= 1    + parts->tm_mon;
31     int day = parts->tm_mday ;
32
33     ofstream myfile;
34     myfile.open("respuesta.txt",ios::out | ios::app);
35     string respues = "";
36     if (myfile.is_open())
37     {
38         for (size_t i=0;i<preguntas.size();i++)
39         {
40             respues+=preguntas[i]->getRespuesta()+"\t";
41         }
42         myfile « 1900 + parts->tm_year  « "_";
43         myfile « 1    + parts->tm_mon   « "_";
44         myfile «        parts->tm_mday  « "\t";
45         myfile « respues «"\n";
46         myfile.close();
47         return true;
48     }
49     else cout « "No se puede guardar";
50     return false;
51 }
```

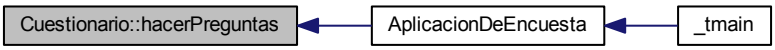
Gráfico de llamadas a esta función:

4.1.3.3. void Cuestionario::hacerPreguntas ( )

Definición en la línea 53 del archivo Cuestionario.cpp.

```
54 {
55     int numeroPreguntas = this->preguntas.size();
56 //  cout«preguntas.size()«"eltama~no\n";
57     for (int contadorPreguntas=0;contadorPreguntas<numeroPreguntas;contadorPreguntas++)
58     {
59         preguntas[contadorPreguntas]->imprimirPregunta();
60         preguntas[contadorPreguntas]->responder();
61     }
62 }
```

Gráfico de llamadas a esta función:



La documentación para esta clase fue generada a partir de los siguientes ficheros:

- ProyectoIntegrador2/**Cuestionario.h**
- ProyectoIntegrador2/**Cuestionario.cpp**

4.2. Referencia de la Clase Encriptador

`#include <Encriptador.h>`

Métodos públicos

- **Encriptador** (std::string claveE, std::string claveD)
- ~**Encriptador** ()
- std::string **Cifrar** (std::string input)
- std::string **Decifrar** (std::string input)

4.2.1. Descripción detallada

Definición en la línea 16 del archivo Encriptador.h.

4.2.2. Documentación del constructor y destructor

4.2.2.1. Encriptador::Encriptador (std::string *claveE,* std::string *claveD*)

Definición en la línea 51 del archivo Encriptador.cpp.

```
52 {
53     if (claveE.length() != claveD.length())
54     {
55         return;
56     }
57     claveEncriptacion=claveE;
58     claveDesencriptacion=claveD;
59     tamanoClave = claveD.length()-1;
60 }
```

4.2.2.2. Encriptador::~Encriptador ( )

Definición en la línea 46 del archivo Encriptador.cpp.

```
47 {
48
49 }
```

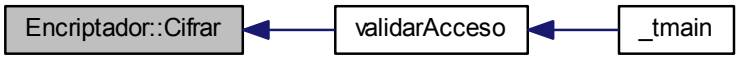
4.2.3. Documentación de las funciones miembro

4.2.3.1. std::string Encriptador::Cifrar (std::string *input*)

Definición en la línea 29 del archivo Encriptador.cpp.

```
30 {
31     for (size_t i =0;i<input.length();i++)
32     {
33         int posE = claveDesencriptacion.find(input[i]);
34         int posF = posE+i;
35         if (posF>tamanoClave) posF = ((posE+i)%tamanoClave)-1;
36         if(posE != std::string::npos)input[i]=claveEncriptacion[posF];
37         else input[i]='#';
38         //std::cout<<posF<<"<Pf Tc>"<<tamanoClave<<std::endl;
39     }
40
41
42
43     return input;
44 }
```

Gráfico de llamadas a esta función:



4.2.3.2. std::string Encriptador::Decifrar (std::string *input*)

Definición en la línea 10 del archivo Encriptador.cpp.

```
11 {
12     for (size_t i =0;i<input.length();i++)
13     {
```

```
14          int posE = claveEncriptacion.find(input[i]);
15          int posF = posE - i;
16          if (posF<0)
17          {
18              posF %= tamanoClave;
19              posF += tamanoClave;
20              posF++;
21          }
22          if(posE != std::string::npos)input[i]=claveDesencriptacion[posF];
23          else input[i]='#';
24          //std::cout<<posF<<"<Pf Tc>"<<tamanoClave<<std::endl;
25      }
26      return input;
27  }
```

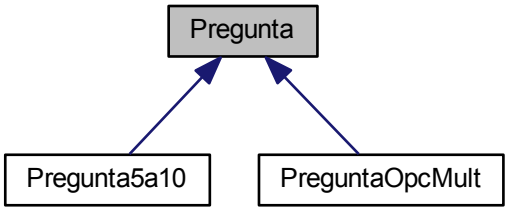
La documentación para esta clase fue generada a partir de los siguientes ficheros:

- ProyectoIntegrador2/**Encriptador.h**
- ProyectoIntegrador2/**Encriptador.cpp**

4.3. Referencia de la Clase Pregunta

`#include <Preguntas.h>`

Diagrama de herencias de Pregunta



Métodos públicos

- **Pregunta** (std::string)
- ~**Pregunta** ()
- virtual void **responder** ()
- virtual void **imprimirPregunta** ()
- std::string **getRespuesta** ()

Atributos protegidos

- char **respuesta** [512]

4.3.1. Descripción detallada

Definición en la línea 6 del archivo Preguntas.h.

4.3.2. Documentación del constructor y destructor

4.3.2.1. Pregunta::Pregunta (std::string *pregunta*)

Definición en la línea 9 del archivo Preguntas.cpp.

```
10 {
11     this->pregunta=pregunta;
12     for (int i = 0; i < 512; i++)
13     {
14         respuesta[i]='\0';
15     }
16 }
```

4.3.2.2. Pregunta::~Pregunta ( ) [inline]

Definición en la línea 10 del archivo Preguntas.h.

```
10 {};
```

4.3.3. Documentación de las funciones miembro

4.3.3.1. std::string Pregunta::getRespuesta ( )

Definición en la línea 41 del archivo Preguntas.cpp.

```
42 {
43     return *(new string(respuesta));
44 }
```

4.3.3.2. void Pregunta::imprimirPregunta ( ) [virtual]

Reimplementado en **PreguntaOpcMult** (p. 17) y **Pregunta5a10** (p. 15).

Definición en la línea 36 del archivo Preguntas.cpp.

```
37 {
38     cout«"\n» "«pregunta«": ";
39 }
```

4.3.3.3. void Pregunta::responder ( ) [virtual]

Reimplementado en **PreguntaOpcMult** (p. 17) y **Pregunta5a10** (p. 15).

Definición en la línea 18 del archivo Preguntas.cpp.

```
19 {
20     //cin.ignore();
21 //  this->imprimirPregunta();
22     cin.getline(respuesta,512);
23     for (int i=0;i<512;i++)
24     {
25         if (respuesta[i]=='\0')
26         {
27             break;
28         }
29         if (respuesta[i]=='\t')
30         {
31             respuesta[i]=' ';
32         }
33     }
34 }
```

4.3.4. Documentación de los datos miembro

4.3.4.1. char Pregunta::respuesta[512] [protected]

Definición en la línea 17 del archivo Preguntas.h.

La documentación para esta clase fue generada a partir de los siguientes ficheros:

- ProyectoIntegrador2/**Preguntas.h**
- ProyectoIntegrador2/**Preguntas.cpp**

4.4. Referencia de la Clase Pregunta5a10

`#include <Preguntas.h>`

Diagrama de herencias de Pregunta5a10

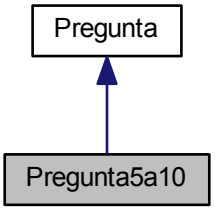
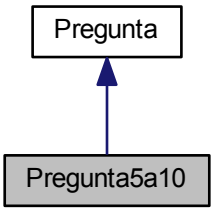


Diagrama de colaboración para Pregunta5a10:



Métodos públicos

- **Pregunta5a10** (std::string)
- ~**Pregunta5a10** ()
- void **responder** ()
- void **imprimirPregunta** ()

Otros miembros heredados

4.4.1. Descripción detallada

Definición en la línea 20 del archivo Preguntas.h.

4.4.2. Documentación del constructor y destructor

4.4.2.1. Pregunta5a10::Pregunta5a10 (std::string *input*)

Definición en la línea 46 del archivo Preguntas.cpp.

```
46                                          : Pregunta(input)
47 {
48
49 }
```

4.4.2.2. Pregunta5a10::~Pregunta5a10 ( ) [inline]

Definición en la línea 24 del archivo Preguntas.h.

```
24 {};
```

4.4.3. Documentación de las funciones miembro

4.4.3.1. void Pregunta5a10::imprimirPregunta ( ) [virtual]

Reimplementado de **Pregunta** (p. 12).

Definición en la línea 66 del archivo Preguntas.cpp.

```
67 {
68     //Pregunta::imprimirPregunta();
69     __super::imprimirPregunta();
70     cout«" (5 a 10) | ";
71 }
```

4.4.3.2. void Pregunta5a10::responder ( ) [virtual]

Reimplementado de **Pregunta** (p. 12).

Definición en la línea 51 del archivo Preguntas.cpp.

```
52 {
53 //  imprimirPregunta();
54 __super::responder();
55 //cout«respuesta«"res\n";
56 istringstream ss(respuesta);
57 int x;
58 if (!(ss » x) || (x>10 || x<5) || respuesta[2]!='\0') {
59
60
61     cout«"Numero Invalido\n";
62     responder();
63 }
64 }
```

La documentación para esta clase fue generada a partir de los siguientes ficheros:

- ProyectoIntegrador2/**Preguntas.h**
- ProyectoIntegrador2/**Preguntas.cpp**

4.5. Referencia de la Clase PreguntaOpcMult

`#include <Preguntas.h>`

Diagrama de herencias de PreguntaOpcMult

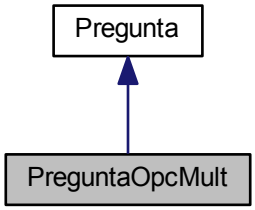
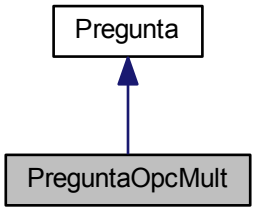


Diagrama de colaboración para PreguntaOpcMult:



Métodos públicos

- **PreguntaOpcMult** (std::string, std::string)
- ~**PreguntaOpcMult** ()
- void **responder** ()
- void **imprimirPregunta** ()

Otros miembros heredados

4.5.1. Descripción detallada

Definición en la línea 30 del archivo Preguntas.h.

4.5.2. Documentación del constructor y destructor

4.5.2.1. PreguntaOpcMult::PreguntaOpcMult (std::string *input,* std::string *opci*)

Definición en la línea 73 del archivo Preguntas.cpp.

```
73                                                                  : Pregunta(input)
74 {
75     //opciones=opci;
76     size_t noOpc;
77     numeroOpciones=0;
78     string container="";
79     for (noOpc=0;noOpc<opci.length();noOpc++)
80     {
81         if (opci[noOpc]=='\t' || noOpc==(opci.length()-1))
82         {
83             numeroOpciones++;
84             opcions.push_back(container);
85             container="";
86             continue;
87         }
88         container+=opci[noOpc];
89     }
90     //cout«opci.length()«"lasopcios\n";
91     //numeroOpciones=opci.length();
92 }
```

4.5.2.2. PreguntaOpcMult::~PreguntaOpcMult ( ) [inline]

Definición en la línea 34 del archivo Preguntas.h.

```
34 {};
```

4.5.3. Documentación de las funciones miembro

4.5.3.1. void PreguntaOpcMult::imprimirPregunta ( ) [virtual]

Reimplementado de **Pregunta** (p. 12).

Definición en la línea 115 del archivo Preguntas.cpp.

```
116 {
117     __super::imprimirPregunta();
118     cout«endl;
119 
120     for (int i=0;i<numeroOpciones;i++)
121     {
122         cout«char(i+97)«") "«opcions[i]«"\t";
123     }
124     cout«"\n";
125 }
```

4.5.3.2. void PreguntaOpcMult::responder ( ) [virtual]

Reimplementado de **Pregunta** (p. 12).

Definición en la línea 94 del archivo Preguntas.cpp.

```
95 {
96     string opciones="";
97     for (int j = 0; j < numeroOpciones; j++)
98     {
99         opciones+=char(j+97);
100     }
101     char opc = _getch();
102     for (unsigned int i=0;i<opciones.length();i++)
103     {
104         if (opciones[i] == opc)
105         {
106             respuesta[0]=opc;
107             return;
108         }
109 
110     }
111     responder();
112 
113 }
```

La documentación para esta clase fue generada a partir de los siguientes ficheros:

- ProyectoIntegrador2/**Preguntas.h**
- ProyectoIntegrador2/**Preguntas.cpp**

4.6. Referencia de la Clase Validador

```
#include <Encriptador.h>
```

Métodos públicos

- **Validador** (std::string filename)
- ~**Validador** ()
- bool **esValido** (std::string input)
- std::string **getContent** ()

Atributos públicos

- std::string **path**
- int **size**

4.6.1. Descripción detallada

Definición en la línea 31 del archivo Encriptador.h.

4.6.2. Documentación del constructor y destructor

4.6.2.1. Validador::Validador (std::string *filename*)

Definición en la línea 86 del archivo Encriptador.cpp.

```
87 {
88     for (char i='A';i<='Z';i++)
89     {
90         //std::cout«i«std::endl;
91         std::string tmpcontent;
92         path = "N:\\"+filename;
93         path[0]=i;
94         //std::cout«"Checando en "+path«std::endl;
95         std::ifstream archivoClave (path, std::ifstream::binary);
96         if (archivoClave.is_open())
97         {
98
99             content.assign( (std::istreambuf_iterator<char>(archivoClave) ),
100                  (std::istreambuf_iterator<char>()    ) );
101             break;
102             tmpcontent.assign(content.data());
103         }
104
105         //std::cout«tmpcontent«std::endl;
106
107     }
108 }
```

4.6.2.2. Validador::~Validador ( )

Definición en la línea 81 del archivo Encriptador.cpp.

```
82 {
83
84 }
```

4.6.3. Documentación de las funciones miembro

4.6.3.1. bool Validador::esValido (std::string *input*)

Definición en la línea 68 del archivo Encriptador.cpp.

```
69 {
70     std::string contenido = getContent();
71     if (input == contenido)
72     {
73         return true;
74     }
75     else
76     {
77         return false;
78     }
79 }
```

Gráfico de llamadas para esta función:

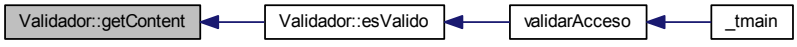
Gráfico de llamadas a esta función:

4.6.3.2. std::string Validador::getContent ( )

Definición en la línea 63 del archivo Encriptador.cpp.

```
64 {
65     return content;
66 }
```

Gráfico de llamadas a esta función:



4.6.4. Documentación de los datos miembro

4.6.4.1. std::string Validador::path

Definición en la línea 37 del archivo Encriptador.h.

4.6.4.2. int Validador::size

Definición en la línea 38 del archivo Encriptador.h.

La documentación para esta clase fue generada a partir de los siguientes ficheros:

- ProyectoIntegrador2/**Encriptador.h**
- ProyectoIntegrador2/**Encriptador.cpp**

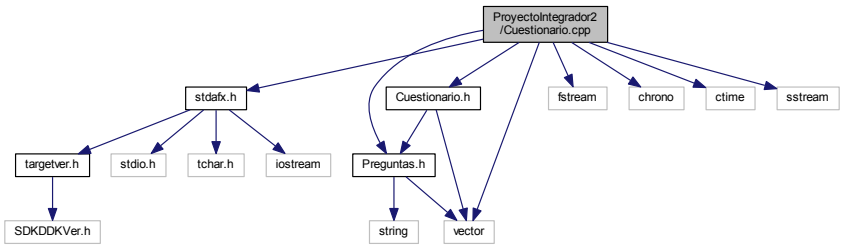
Capítulo 5

Documentación de archivos

5.1. Referencia del Archivo ProyectoIntegrador2/Cuestionario.cpp

```
#include "stdafx.h"
#include "Cuestionario.h"
#include "Preguntas.h"
#include <vector>
#include <fstream>
#include <chrono>
#include <ctime>
#include <sstream>
```

Dependencia gráfica adjunta para Cuestionario.cpp:



5.2. Referencia del Archivo ProyectoIntegrador2/Cuestionario.h

```
#include "Preguntas.h"
#include <vector>
```

Dependencia gráfica adjunta para Cuestionario.h:

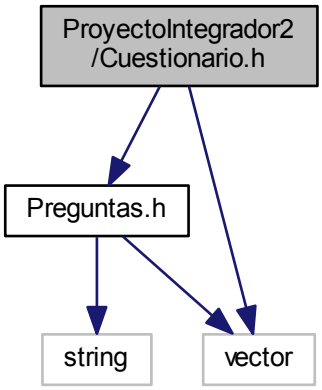
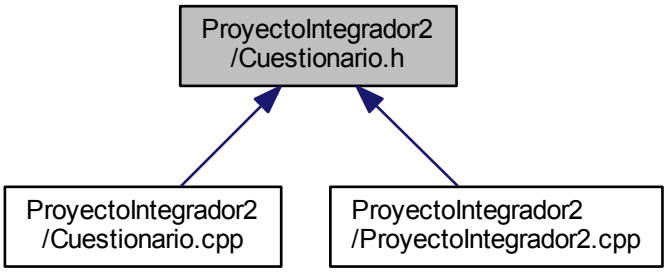


Gráfico de los archivos que directa o indirectamente incluyen a este archivo:



Clases

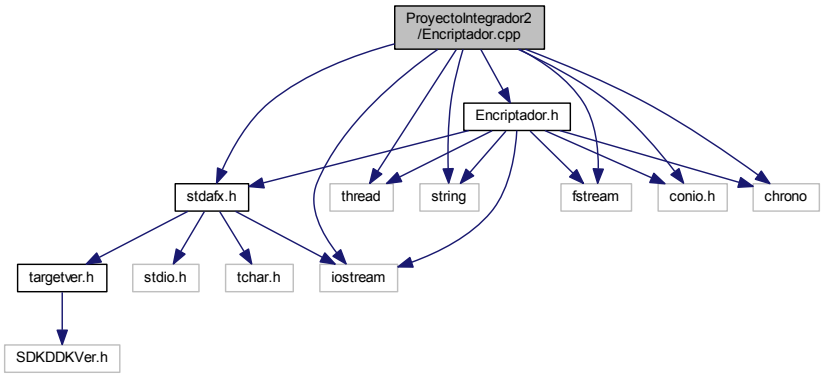
- class **Cuestionario**

5.3. Referencia del Archivo ProyectoIntegrador2/Encriptador.cpp

```
#include "stdafx.h"
```

```
#include <thread>
#include <string>
#include <iostream>
#include <fstream>
#include <conio.h>
#include <chrono>
#include "Encriptador.h"
```

Dependencia gráfica adjunta para Encriptador.cpp:



Funciones

- bool **validarAcceso** ()

5.3.1. Documentación de las funciones

5.3.1.1. bool validarAcceso ( )

Definición en la línea 110 del archivo Encriptador.cpp.

```
111 {
112     Encriptador crypt("RSeCDf23g4ihjlk6auvMLxpQUmAEBwzyFGontJHIqKr7 Ts8ON9PXVWZ0Y15bcd", "
    abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789 ");
113     Validador valid("clave.txt");
114     string clave = "pablito clavo un clavito en la calva de un calvito";
115     //cout«"Buscando el archivo"«endl;
116     //if(valid.encontrado)cout«"Archivo encontrado en " « valid.path«endl;
117     //cout«crypt.Cifrar(clave)«endl;
118     bool validacion =valid.esValido(crypt.Cifrar(clave));
119     if (validacion)
120     {
121         //      Acceso=true;
122         //cout«"Acceso Permitido"«endl;
123         return true;
124     }
125     else
126     {
127         //      Acceso=false;
128         cout«"Acceso Denegado"«endl;
129 
130         return false;
131 
132     }
133     //cout«"Contenido del archivo: "«valid.getContent()«endl;
```

```
134     //  cout«"Contenido del archivo decifrado: " « crypt.Decifrar(valid.content)«endl;
135     //  cout«"Clave correcta: " « clave«endl;
136 }
```

Gráfico de llamadas para esta función:

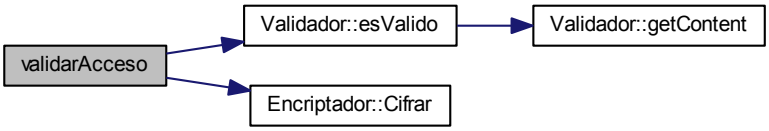
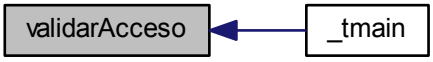


Gráfico de llamadas a esta función:



5.4. Referencia del Archivo ProyectoIntegrador2/Encriptador.h

```
#include "stdafx.h"
#include <thread>
#include <string>
#include <iostream>
#include <fstream>
#include <conio.h>
#include <chrono>
```

Dependencia gráfica adjunta para Encriptador.h:

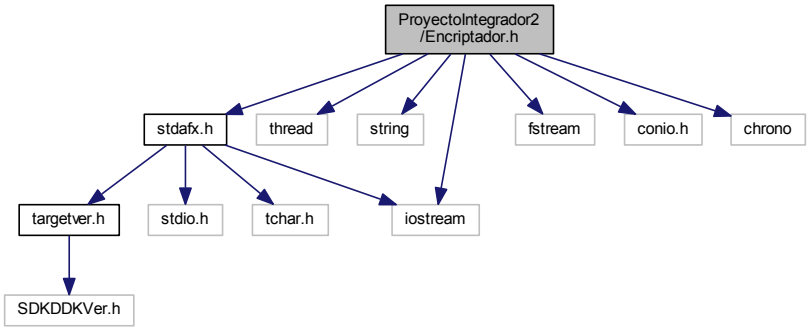
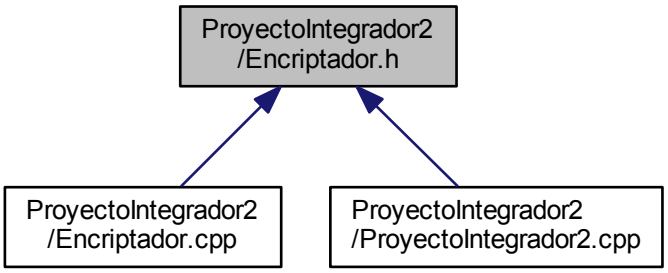


Gráfico de los archivos que directa o indirectamente incluyen a este archivo:



Clases

- class **Encriptador**
- class **Validador**

Funciones

- bool **validarAcceso** ()

5.4.1. Documentación de las funciones

5.4.1.1. bool validarAcceso ( )

Definición en la línea 110 del archivo Encriptador.cpp.

```
111 {
112     Encriptador crypt("RSeCDf23g4ihjlk6auvMLxpQUmAEBwzyFGontJHIqKr7 Ts8ON9PXVWZ0Y15bcd", "
    abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789 ");
113     Validador valid("clave.txt");
114     string clave = "pablito clavo un clavito en la calva de un calvito";
115     //cout«"Buscando el archivo"«endl;
116     //if(valid.encontrado)cout«"Archivo encontrado en " « valid.path«endl;
117     //cout«crypt.Cifrar(clave)«endl;
118     bool validacion =valid.esValido(crypt.Cifrar(clave));
119     if (validacion)
120     {
121         //      Acceso=true;
122         //cout«"Acceso Permitido"«endl;
123         return true;
124     }
125     else
126     {
127         //      Acceso=false;
128         cout«"Acceso Denegado"«endl;
129 
130         return false;
131 
132     }
133     //cout«"Contenido del archivo: "«valid.getContent()«endl;
134     //  cout«"Contenido del archivo decifrado: " « crypt.Decifrar(valid.content)«endl;
135     //  cout«"Clave correcta: " « clave«endl;
136 }
```

Gráfico de llamadas para esta función:

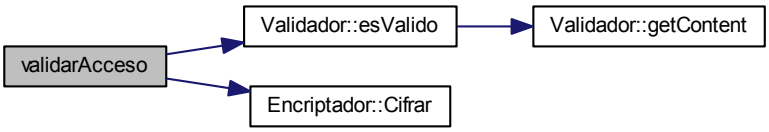
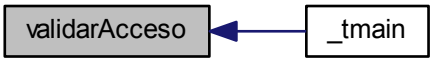


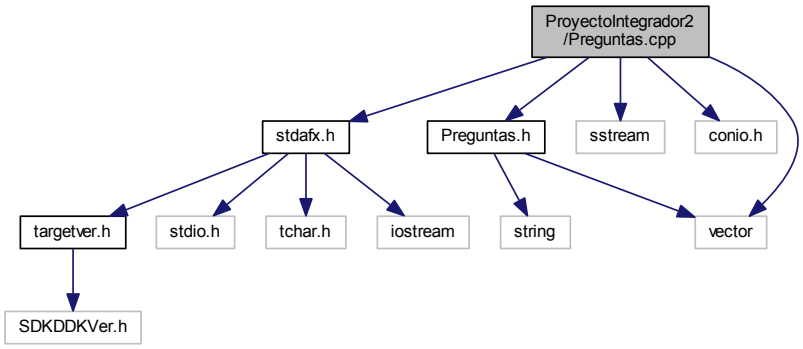
Gráfico de llamadas a esta función:



5.5. Referencia del Archivo ProyectoIntegrador2/Preguntas.cpp

```
#include "stdafx.h"
#include "Preguntas.h"
#include <sstream>
#include <conio.h>
#include <vector>
```

Dependencia gráfica adjunta para Preguntas.cpp:



5.6. Referencia del Archivo ProyectoIntegrador2/Preguntas.h

```
#include <string>
#include <vector>
```

Dependencia gráfica adjunta para Preguntas.h:

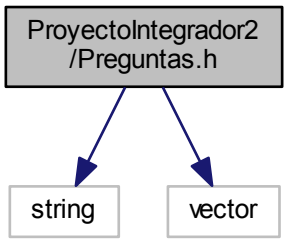
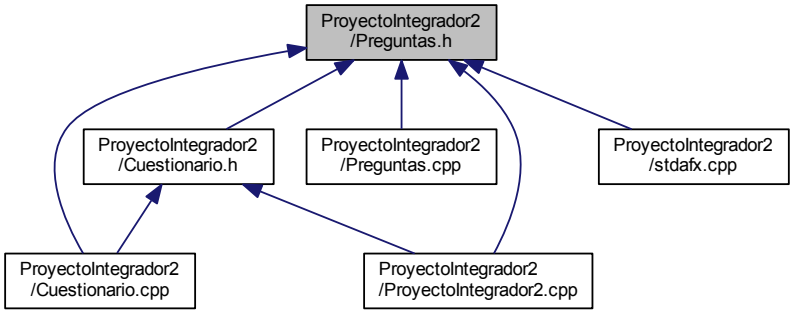


Gráfico de los archivos que directa o indirectamente incluyen a este archivo:



Clases

- class **Pregunta**
- class **Pregunta5a10**
- class **PreguntaOpcMult**

5.7. Referencia del Archivo ProyectoIntegrador2/ProyectoIntegrador2.cpp

```
#include "stdafx.h"
#include <thread>
#include <iostream>
#include <chrono>
#include "Encriptador.h"
#include "Preguntas.h"
#include "Cuestionario.h"
```

Dependencia gráfica adjunta para ProyectoIntegrador2.cpp: