# cs225f20-b.sg

Dashboard / My courses / cs225f20-b.sg / 5 October - 11 October / Assignment 3 upload

Description Submission Edit Submission view

## Grade

Reviewed on Tuesday, 13 October 2020, 9:20 PM by Automatic grade
**grade**: A

**Assessment report**[-]

[-]**Summary of tests**

```
+------------------------------+
| 71 tests run/71 tests passed |
+------------------------------+
```

Submitted on Tuesday, 13 October 2020, 9:20 PM (Download)

## Constructor.cpp

```cpp
#include "Constructor.h"
#include "Text.h"
#include "Html.h"
#include "Markdown.h"

std::vector<std::string> get_options()
{
    std::vector < std::string > v;

    v.push_back("Text");
    v.push_back("Markdown");
    v.push_back("HTML");

    return v;
}

IFormatterFactory* construct(int selected_option)
{
    IFormatterFactory* result = NULL;

    switch (selected_option)
    {
        case 0 : result = new Text(); break;
        case 1 : result = new Markdown(); break;
        case 2: result = new Html(); break;
        default: break;
    }

    return result;
}

void save(ISavable& savable, std::ostream& stream)
{
    savable.save(stream);
}

```

## Html.h

```
#ifndef _HTML_H_
#define _HTML_H_

#include "IFormatterFactory.h"
#include "ISavable.h"

class Html : public IFormatterFactory, public IFormatterElement, public ISavable
{
public:
    std::string str;

    Html() = default;
    Html(std::string s): str{s}{};

    IFormatterElement* create_begin() override
    {
        std::string newString;

        newString += "<!doctype html>\n";
        newString += "<html lang=en>\n";
        newString += "<style>blockquote { border-left: 4px solid #d0d0d0;";
        newString += " padding: 4px; }</style>\n";
        newString += "<head>\n";
        newString += "\t<meta charset=utf-8>\n";
        newString += "\t<title>Result</title>\n";
        newString += "</head>\n";
        newString += "<body>\n";

        return new Html(newString);
    }

    IFormatterElement* create_header1(std::string s) override
    {
        std::string newString;

        newString += "\t<h1>1. ";
        newString += s;
        newString += "</h1>\n";

        return new Html{newString};
    }

    IFormatterElement* create_paragraph(std::string s) override
    {
        std::string newString;

        newString += "\t<p>";
        newString += s;
        newString += "</p>\n";

        return new Html{newString};
    }

    IFormatterElement* create_header2(std::string s) override
    {
        std::string newString;

        static int count = 0;
        ++count;

        newString += "\t<h2>";

        if (count == 1)
        {
            newString += "1.1. ";
        }
        else
        {
            newString += "1.2. ";
        }

        newString += s;
        newString += "</h2>\n";

        return new Html{newString};
    }

    IFormatterElement* create_blockquote(std::string s) override
    {
        std::string newString;

        newString +="\t<blockquote>";
        newString += s;
        newString += "</blockquote>\n";

        return new Html{newString};
    }

    IFormatterElement* create_end()  override
    {
        std::string newString;

        newString += "</body>\n";
        newString += "</html>\n";

        return new Html{newString};
    }

    void save(std::ostream& stream)
    {
        stream << str;
    }

    ~Html(){}
};

#endif // _HTML_H_
```

## Text.h

```cpp
#ifndef _TEXT_H_
#define _TEXT_H_

#include "IFormatterFactory.h"
#include "ISavable.h"

class Text : public IFormatterFactory, public IFormatterElement, public ISavable
{
public:
    std::string str;

    Text() = default;
    Text(std::string s): str{s}{};

    IFormatterElement* create_begin() override
    {
        std::string newString;
        return new Text(newString);
    }

    IFormatterElement* create_header1(std::string s) override
    {
        std::string newString;

        newString += s;
        newString += "\n";

        return new Text(newString);
    }

    IFormatterElement* create_paragraph(std::string s) override
    {
        std::string newString;

        newString += s;
        newString += "\n";

        static int count = 0;
        ++count;

        if(count<=2)
            newString +="\n";

        return new Text(newString);
    }

    IFormatterElement* create_header2(std::string s) override
    {
        std::string newString;

        newString += s;
        newString += "\n";

        return new Text(newString);
    }

    IFormatterElement* create_blockquote(std::string s) override
    {
        std::string newString;

        newString += "\"";
        newString += s;
        newString += "\"";
        newString += "\n";

        return new Text(newString);
    }

    IFormatterElement* create_end() override
    {
        std::string newString;
        return new Text(newString);
    }

    void save(std::ostream& stream)
    {
        stream << str;
    }

    ~Text() {}
};

#endif //  TEXT H
```

## Markdown.h

```
#ifndef _MARKDOWN_H_
#define _MARKDOWN_H_

#include "IFormatterFactory.h"
#include "ISavable.h"

class Markdown : public IFormatterFactory, public IFormatterElement
, public ISavable
{
public:
     std::string str;

    Markdown() = default;
    Markdown(std::string s): str{s}{};

    IFormatterElement* create_begin()
    {
        std::string newString;

        newString += "\n";
        return new Markdown(newString);
    }

    IFormatterElement* create_header1(std::string s) override
    {
        std::string newString;

        newString += "# 1. ";
        newString += s;
        newString += "\n";

        return new Markdown{newString};
    }

    IFormatterElement* create_paragraph(std::string s) override
    {
        std::string newString;

        newString += s;
        newString += "\n";

        static int count = 0;
        ++count;

        if (count <= 2)
        {
            newString+= "\n";
        }

        return new Markdown{newString};
    }

    IFormatterElement* create_header2(std::string s) override
    {
        std::string newString;

        static int counter = 0;
        ++counter;

        if (counter == 1)
        {
            newString += "## 1.1. ";
        }
        else
        {
            newString += "## 1.2. ";
        }

        newString += s;
        newString += "\n";

        return new Markdown{newString};
    }

    IFormatterElement* create_blockquote(std::string s) override
    {
        std::string newString;

        newString += "> ";
        newString += s;
        newString += "\n\n";

        return new Markdown{newString};
    }

    IFormatterElement* create_end() override
    {
        std::string newString;
        return new Markdown{newString};
    }

    void save(std::ostream& stream)
    {
        stream << str;
    }

    ~Markdown() {}
};

#endif // _MARKDOWN_H_

```

## IFormatterFactory.h

```cpp
#ifndef _IFORMATTERFACTORY_H_
#define _IFORMATTERFACTORY_H_

#include <string>
#include "IFormatterElement.h"

class IFormatterFactory
{
public:
    virtual IFormatterElement* create_begin() = 0;
    virtual IFormatterElement* create_header1(std::string s) = 0;
    virtual IFormatterElement* create_paragraph(std::string s) = 0;
    virtual IFormatterElement* create_header2(std::string s) = 0;
    virtual IFormatterElement* create_blockquote(std::string s) = 0;
    virtual IFormatterElement* create_end() = 0;
    virtual ~IFormatterFactory() = default;
};

#endif // _IFORMATTERFACTORY_H_
```

## IFormatterElement.h

```cpp
#ifndef _IFORMATTERELEMENT_H_
#define _IFORMATTERELEMENT_H_

#include "ISavable.h"

class IFormatterElement
{
public:
    virtual ~IFormatterElement() = default;
};

#endif // _IFORMATTERELEMENT_H_
```

VPL

◄ Assignment 3 files

Jump to... ⬍

Attendance cs225f20-b.sg Tuesday 13/10/2020 1:30pm-3:10pm ►