



Template Method Pattern

Presenters:

Yuxuan Xie
Shashank Singh
Nishant Agrawal





Peter Smith
Digital Marketing Manager

peters@gmail.com
+27(0) 72 876 1243
801010 2002 06 0
Code 10
English, Afrikaans, Zulu
34 Longer Street, Sandton, Johannesburg

EDUCATION

UNIVERSITY OF JOHANNESBURG 2007 - 2005 INFORMATION TECHNOLOGY (WEB DEVELOPMENT)

- Digital Lifestyles
- Computer Security and Privacy
- Productivity Programmes
- The Internet and the World Wide Web
- Computer Basics, PHP, HTML

ST JOHN HIGH SCHOOL 1995 - 1999 MATRIC

- Subjects:
- Maths, Science, English, Afrikaans, Biology, Computers

WORK EXPERIENCE

TOP MEDIA 2006 - 2008 GRAPHICS DESIGNER

Role Summary

- Meeting clients or account managers to discuss the business objectives and requirements of the job;
- Estimating the time required to complete the work and providing quotes for clients;
- Developing design briefs that suit the client's purpose;
- Thinking creatively to produce new ideas and concepts and developing interactive design;
- Using innovation to redefine a design brief within the constraints of cost and time;
- Presenting finalised ideas and concepts to clients or account managers;
- Working with a range of media, including computer-aided design (CAD) and keeping up to date with emerging technologies;
- Proofreading to produce accurate and high-quality work;
- Demonstrating illustrative skills with rough sketches and working on layouts ready for print;
- Commissioning illustrators and photographers;
- Working as part of a team with printers, copywriters, photographers, stylists, illustrators, other designers, account executives, web developers and marketing specialists.

ABOUT ME

I am hard working, self-motivated and a devoted married man with 2 children. I am well experienced with 10 years in web development and digital branding. I have worked in various companies as a project manager and currently studying advance project management degree with UNISA. I am passionate about web development and creative in my approach.

ATTRIBUTES

- MANAGEMENT
- LEADERSHIP
- INNOVATIVE
- FAST LEARNER
- CREATIVE
- MOTIVATED
- DEDICATED
- INDEPENDENT

SKILLS

- | | |
|-----------------|-------|
| • HTML 5 | ●●●●○ |
| • PHP | ●●●●○ |
| • ASP.NET | ●●●●○ |
| • VB.NET | ●●●●○ |
| • JAVA | ●●●●○ |
| • PHOTOSHOP CS6 | ●●●●○ |
| • ILLUSTRATOR | ●●●●○ |
| • FLASH | ●●●●○ |
| • MS OFFICE | ●●●●○ |
| • MYSQL DB | ●●●●○ |



Jane Roe

Business Development Manager

Professional Business Developer with more than four years of experience in the business development processes. Involved in product testing, management, and development of new business opportunities.

jane.roe@gmail.com
202-555-0166

New York, USA
in
linkedin.com/in/jane.roe

jane.roe

SKILLS

- SEO
- Public Speaking
- Negotiation
- Teamwork
- Decision Making
- Research & Strategy
- Emotional Intelligence
- Outbound Marketing
- Email Marketing
- Google Analytics
- Sales & Marketing

LANGUAGES

- English
●●●●●
- Spanish
●●●●○
- French
●●●●○

WORK EXPERIENCE

Business Development Manager AirState Solutions

- 09/2014 - 06/2017 New York, USA
- Successfully managed \$2 - 3 million budget projects and successfully achieved the project scheduled goals.
 - Developed and implemented new marketing and sales plans and defined the strategy for the next 5 years.
 - Reviewed constantly the customer feedback and then suggested ways to improve the processes and customer service levels which increased the satisfaction rate from 81% to 95%.
 - Ensured that new clients will grow into a loyal customer base in a specialist niche market by implementing a new loyalty program.

Business Development Assistant AirState Solutions

- 08/2012 - 09/2014 Chicago, USA
- Increased the customer satisfaction rate by 25% by improving the customer service.
 - Planned, supervised, and coordinated daily activity of 3 junior business analysts.
 - Improved the communication with the Marketing department to better understand the competitive position.
 - Directed the creation and implementation of a Business Continuity Plan, and the management of audit programs.

ORGANIZATIONS

- American Management Association (2015 - Present)
- Association of Private Enterprise Education (2014 - Present)
- eBusiness Association (eBA) (2013 - Present)

EDUCATION

MSc in Economics and Business Administration The University of Chicago

09/2008 - 06/2010

HONOURS AND AWARDS

- Jury Member, Venture Cup Entrepreneurship Competition (2016)
Venture Cup USA
- Sales Individual & Business Development Award (2015)
AirState Business Awards
- Excellence in Customer Partnering Award
IES - Institute of Excellence in Sales

Intent

- Define the skeleton of an algorithm in an operation, deferring some steps to subclasses
- Let subclasses redefine certain steps of an algorithm without changing the algorithm's structure

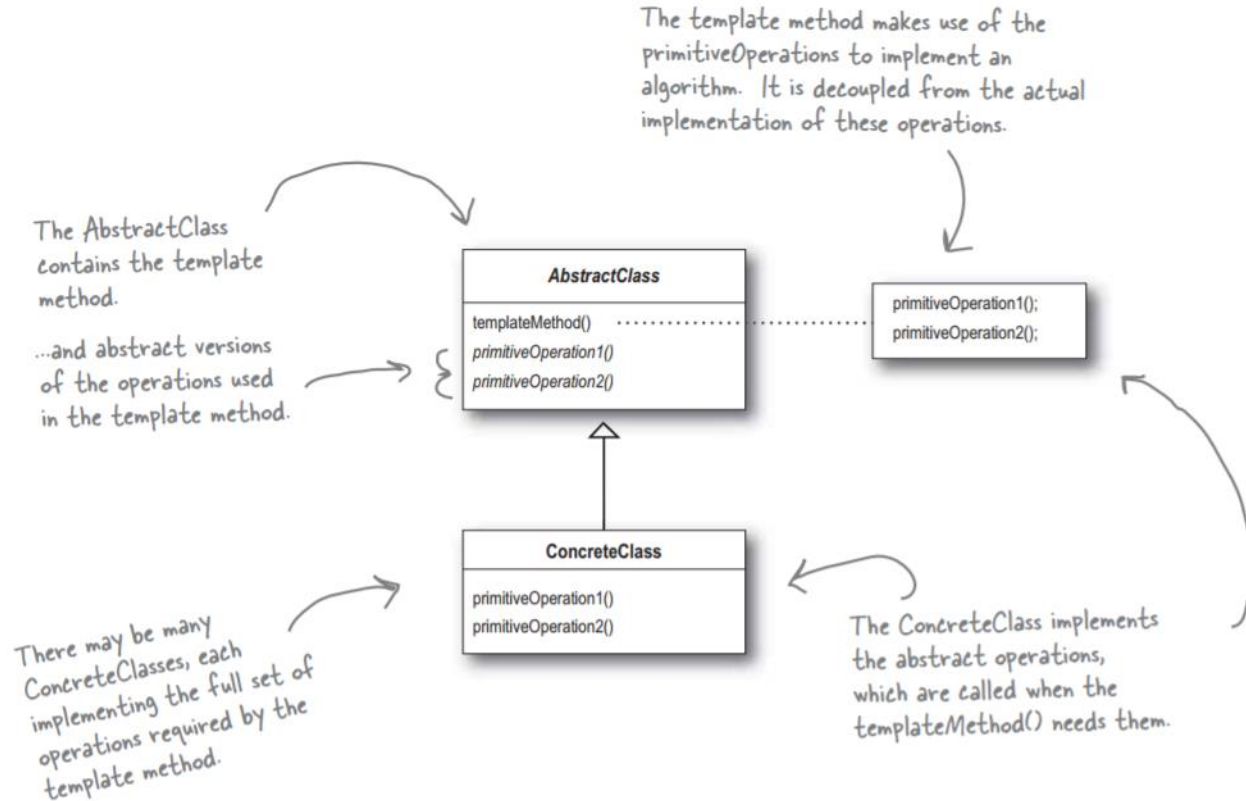
Problem

- How can the invariant parts of a behavior be implemented once so that subclasses can implement the variant parts?
 - Define abstract operations for the variant parts of a behavior.
- How can subclasses redefine certain parts of a behavior without changing the behavior's structure?
 - Define a *template method* that
 - implements the invariant parts of a behavior
 - calls primitive methods that subclasses implement.

Applicability

- to implement the invariant parts of an algorithm once and leave it up to subclasses to implement the behavior that can vary
- when common behavior among subclasses should be factored and localized in a common class to avoid code duplication.
- to control subclasses extensions. You can define a template method that calls “hook” operations at specific points, thereby permitting extensions only at those points.

Structure



Participants

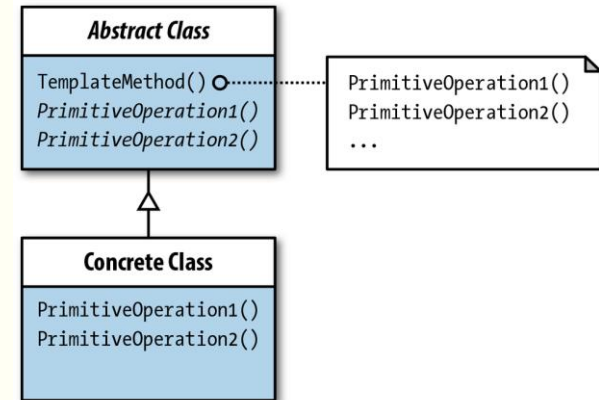
- Abstract Class

Defines a `templateMethod()` operation that defines the skeleton (template) of a behavior by implementing the invariant parts of the behavior and calling abstract `primitive1()` and `primitive2()` operations to defer implementing the variant parts to `SubClass1`.

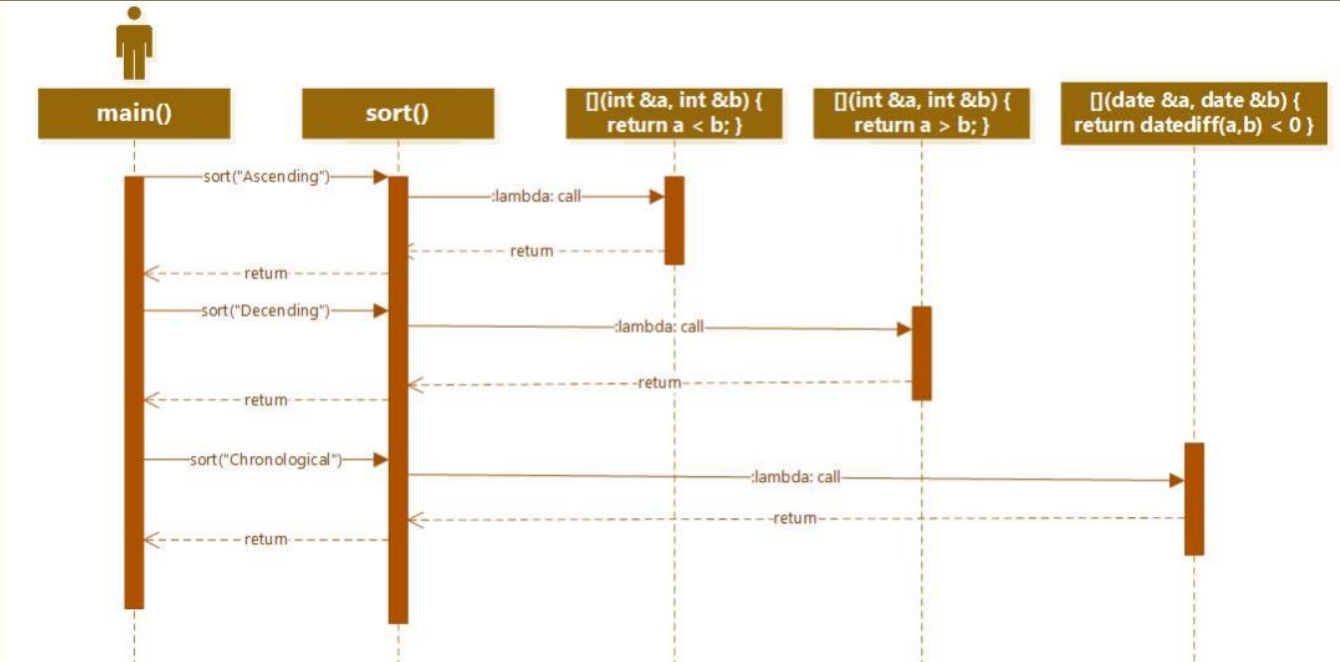
Defines abstract primitive operations for the variant parts of a behavior.

- Concrete Class

Implement the abstract primitive operations.



Collaborations



- ConcreteClass relies on AbstractClass to implement the invariant steps of the algorithm.

Consequences

- Code Reuse

Template methods are a fundamental technique for code reuse. They are particularly important in class libraries because they are the means for factoring out common behavior in library classes.

- Inversion of Control

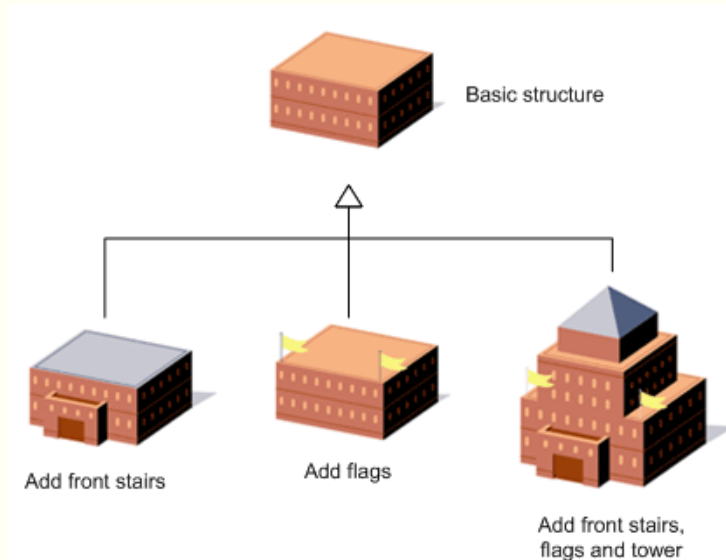
Template methods lead to an inversion of control because subclasses no longer control how the behavior of a parent class is redefined.

IoC = Hollywood Principle

“Don’t call us, we’ll call you.”

Examples

“All the great legends are Templates for human behavior. I would define a myth as a story that has survived.” - John Boorman



Implementation- general example

Framework

Problem: Various steps of this framework used in different applications.

```
//concrete class
class ApplicationOne:public Framework {
public:
    void stepTwo() {
        cout << "Step 2 executing from ApplicationOne\n";
    }
};

//concrete class
class ApplicationTwo :public Framework {
public:
    void stepTwo(){
        cout << "Step 2 executing ApplicationTwo\n";
    }
};
```

```
//abstract class
class Framework {
public:
    void templateMethod() {
        stepOne();
        stepTwo();
        stepThree();
    }

private:
    void stepOne() {
        cout << "Step 1 executing \n";
    }

    virtual void stepTwo() = 0;

    void stepThree() {
        cout << "Step 3 executing \n";
    }
};
```

Implementation- general example

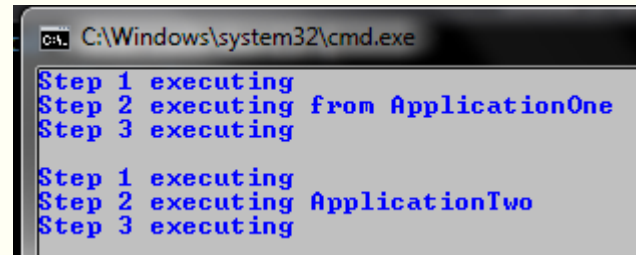
```
int main()
{
    Framework *objects[] =
    {
        new ApplicationOne, new ApplicationTwo
    };

    objects[0]->templateMethod();
    cout << "\n";
    objects[1]->templateMethod();

    cout << "\n\n";
}
```

void * _cdecl operator new(size_t _Size) + 3 overloads

Output:



```
C:\Windows\system32\cmd.exe
Step 1 executing
Step 2 executing from ApplicationOne
Step 3 executing

Step 1 executing
Step 2 executing ApplicationTwo
Step 3 executing
```

Implementation- detailed example

Sorting Algorithms

Problem: Sorting, reversing and displaying selection, insertion and bubble sort algorithms.

```
//Sort Algorithm abstract class
class SortAlgo{
public:
    void execute(int arr[], int n) {
        sort(arr, n);
        if (reverseOrder() == true) {
            displayDesc(arr, n);
        }
        else {
            displayAsc(arr, n);
        }
    }
};
```

```
private:
    virtual void sort(int arr[], int n) = 0;
    virtual bool reverseOrder() { return false; }

    void displayDesc(int arr[], int n) {
        cout << "Descending.....\n";
        int i;
        for (i = n-1; i >= 0; i--)
            printf("%d ", arr[i]);
        printf("\n");
    }

    void displayAsc(int arr[], int n) {
        cout << "Ascending.....\n";
        int i;
        for (i = 0; i < n; i++)
            printf("%d ", arr[i]);
        printf("\n");
    }
};
```

Implementation- detailed example

Selection Sort

```
//Selection Sort
class SelectionSort : public SortAlgo
{
    void sort(int arr[], int n) {
        cout << "\nUsing selection sort.....\n";
        int i, j, min_idx;
        for (i = 0; i < n - 1; i++)
        {
            min_idx = i;
            for (j = i + 1; j < n; j++)
                if (arr[j] < arr[min_idx])
                    min_idx = j;
            swap(&arr[min_idx], &arr[i]);
        }
    }
    void swap(int *xp, int *yp)
    {
        int temp = *xp;
        *xp = *yp;
        *yp = temp;
    }
};
```

Bubble Sort

```
//Bubble Sort
class BubbleSort : public SortAlgo
{
    void sort(int arr[], int n) {
        cout << "\nUsing Bubble sort.....\n";

        int i, j;
        for (i = 0; i < n - 1; i++)
            for (j = 0; j < n - i - 1; j++)
                if (arr[j] > arr[j + 1])
                    swap(&arr[j], &arr[j + 1]);
    }

    void swap(int *xp, int *yp)
    {
        int temp = *xp;
        *xp = *yp;
        *yp = temp;
    }
};
```

Implementation- detailed example

Insertion Sort

```
class InsertionSort : public SortAlgo
{
    void sort(int arr[], int n) {
        cout << "\nUsing Insertion sort.....\n";

        int i, key, j;
        for (i = 1; i < n; i++)
        {
            key = arr[i];
            j = i - 1;
            while (j >= 0 && arr[j] > key)
            {
                arr[j + 1] = arr[j];
                j = j - 1;
            }
            arr[j + 1] = key;
        }

        //hook operation extending this virtual method from base class
        bool reverseOrder() {
            return true;
        }
    }
};
```

Output:

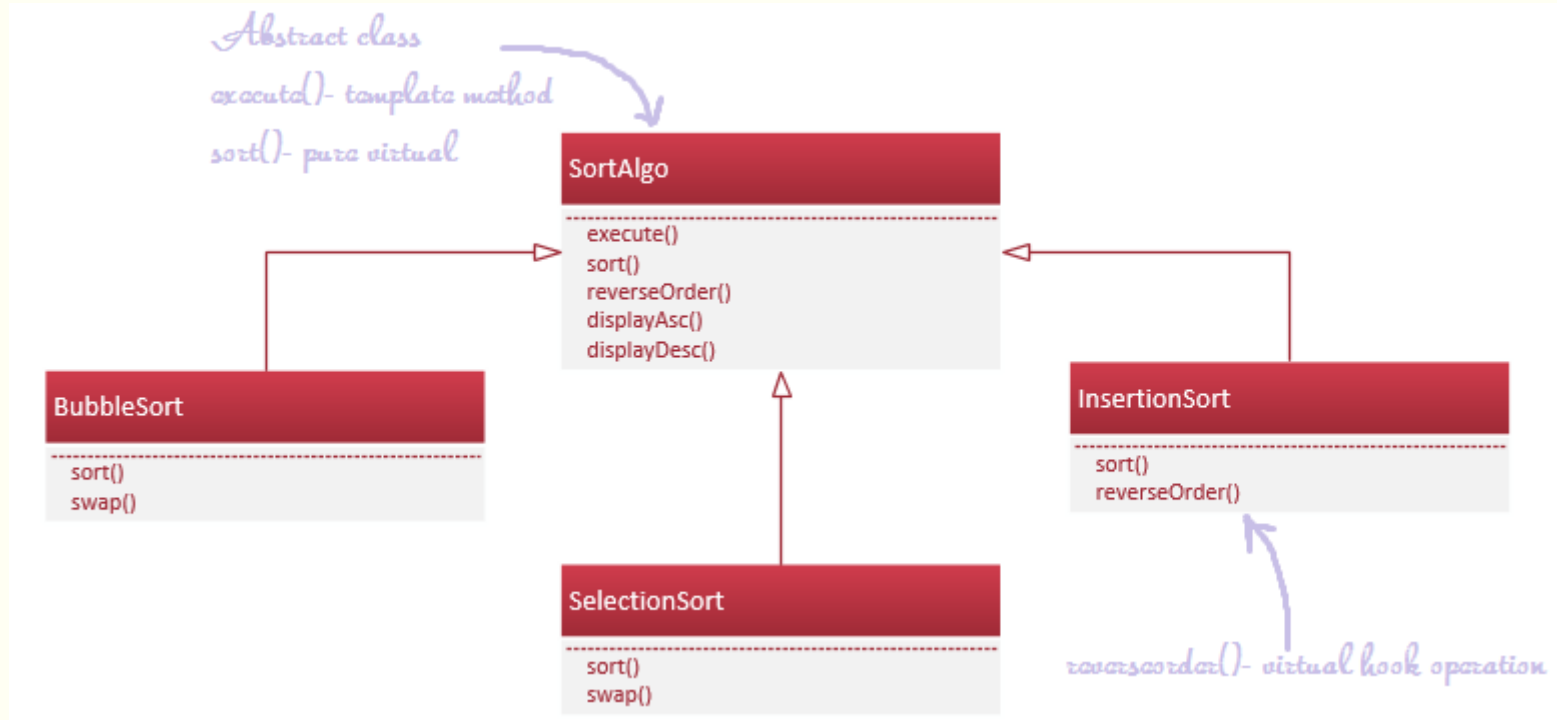
```
Unsorted array.....
2 8 5 1 10 5 9 9 3 5

Using selection sort.....
Ascending.....
1 2 3 5 5 5 8 9 9 10

Using Bubble sort.....
Ascending.....
1 2 3 5 5 5 8 9 9 10

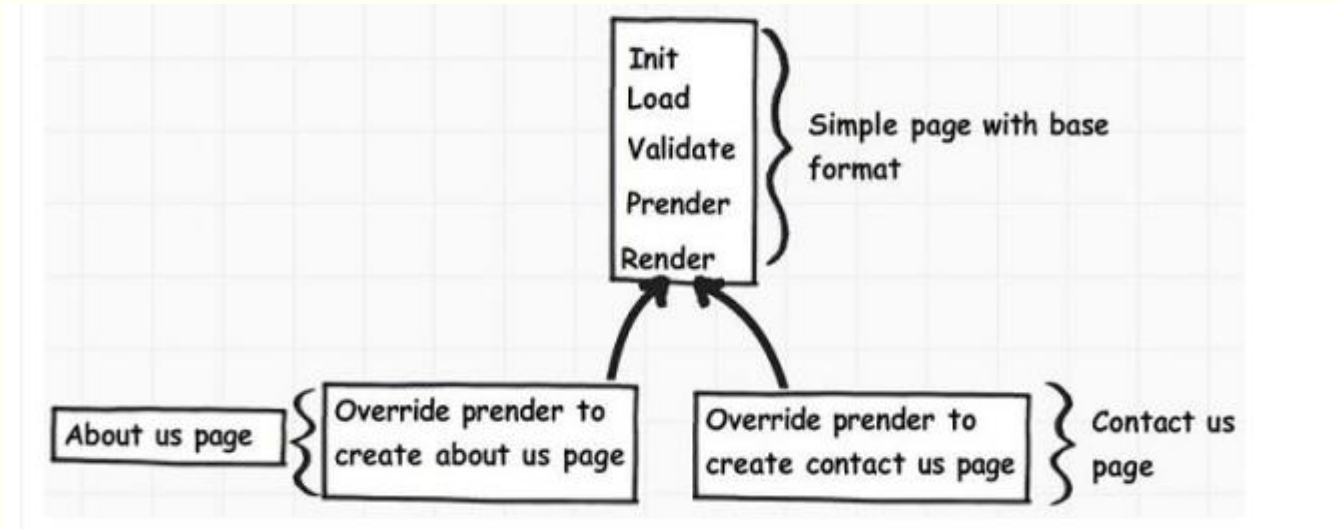
Using Insertion sort.....
Descending.....
10 9 9 8 5 5 5 3 2 1
Press any key to continue . . .
```

Implementation- class diagram



Known Uses:

Can be found in almost all abstract classes.



ASP.net page life cycle

Implementation Issues

1. Access Control

- Primitive/Abstract method - pure virtual and protected/private
- Template Method - non-virtual and public

1. Minimizing primitive operations

1. Naming convention

Related Patterns

- **Factory method**
Specialized template methods
Replace abstract methods
- **Strategy**
Delegation vs inheritance
Modifies logic of objects instead of class

Reference

<https://www.oreilly.com/library/view/learning-php-design/9781449344900/ch09.html>

<http://ramesh-java-design-patterns.blogspot.com/2017/12/template-method-design-pattern.html>

<http://w3sdesign.com/index0100.php#gf>

https://www.google.com/search?q=resume&rlz=1C1CHBF_enUS812US812&source=lnms&tbm=isch&sa=X&ved=0ahUKEwjwm66f7ZXeAhWpмуAKHY_sABwQ_AUIDigB&biw=1280&bih=648#imgdii=eeHiGsOq7Lu16M:&imgsrc=okaiq7zkmpcnnM:

Thank You

Implementing this pattern will make your code *crisp* and your co-workers *jealous*.

Questions??