namespace task的构造函数

{

class Program

{

public static void task01\_method()

{

Console.WriteLine("我是无参无返回值的方法");

}

public static int task08\_method()

{

return 9421;

}

public static int task10\_method(double raw)

{

return (int)raw;

}

public static void task04\_method(string name)

{

Console.WriteLine("我是有参无返回值的方法");

}

static void Main(string[] args)

{

/\*▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓

\* task的构造函数

\*▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓

\*/

//━┅━┅━┅━┅━┅━┅━┅━┅━━┅━┅━┅━┅━┅━┅━┅━

//无返回值无参的方法

//━┅━┅━┅━┅━┅━┅━┅━┅━━┅━┅━┅━┅━┅━┅━┅━

//---------------------方法-------------------------

//Task(Action action)

Task task01 = new Task(new Action(task01\_method));

task01.Start(); task01.Wait();

//-------------------lambda表达式---------------------

Task task02 = new Task(new Action(() => { Console.WriteLine("我是无参无返回值的方法"); }));

task02.Start(); task02.Wait();

Task task03 = new Task(() => { Console.WriteLine("我是无参无返回值的方法"); });

task03.Start(); task03.Wait();

//━┅━┅━┅━┅━┅━┅━┅━┅━━┅━┅━┅━┅━┅━┅━┅━

//有参无返回值的方法

//━┅━┅━┅━┅━┅━┅━┅━┅━━┅━┅━┅━┅━┅━┅━┅━

//-----------------------方法--------------------------

Task task04 = new Task(new Action(()=> { task04\_method("Because of you"); }));

task04.Start(); task04.Wait();

//-----------------lambda表达式--------------------------

//Task(new Action<object> action,object obj)

//action是任务要执行的方法，obj是其形参

//使用ArrayList包装所有形参

Task task05 = new Task(new Action<object>((param) => {

ArrayList pList = param as ArrayList;

Console.WriteLine(pList[0]);

Console.WriteLine(pList[1]);

}), new ArrayList() {"Jack",22,"Europe"});

task05.Start(); task05.Wait();

//使用类封装形参

Params p = new Params();

Task task06 = new Task(new Action<object>((\_p)=> {

Params pp = \_p as Params;

Console.WriteLine(pp.name);

Console.WriteLine(pp.age);

Console.WriteLine(pp.city);

}),new Params());

task06.Start(); task06.Wait();

//使用lambda表达式能访问外部变量的特性

string name = "Titanic";

int date = 1912;

Task task07 = new Task(() => {

Console.WriteLine(name);

Console.WriteLine(date);

});

task07.Start(); task07.Wait();

//━┅━┅━┅━┅━┅━┅━┅━┅━━┅━┅━┅━┅━┅━┅━┅━

//有返回值无参的方法

//━┅━┅━┅━┅━┅━┅━┅━┅━━┅━┅━┅━┅━┅━┅━┅━

//----------------------方法-----------------------

Task<int> task08 = new Task<int>(new Func<int>(task08\_method));

task08.Start();

Console.WriteLine(task08.Result);

Task<int> task09 = new Task<int>(new Func<int>(() => { return 9421; }));

task09.Start();

Console.WriteLine(task09.Result);

//━┅━┅━┅━┅━┅━┅━┅━┅━━┅━┅━┅━┅━┅━┅━┅━

//有返回值有参的方法

//━┅━┅━┅━┅━┅━┅━┅━┅━━┅━┅━┅━┅━┅━┅━┅━

//-----------------------方法----------------------------------

Task<int> task10 = new Task<int>(new Func<int>(()=> {return task10\_method(3.14);}));

task10.Start();

Console.WriteLine(task10.Result);

Task<int> task11 = new Task<int>(new Func<object, int>((param)=> { ArrayList pList = param as ArrayList;

return (int)pList[0];

}),new ArrayList() {3.14});

string str = "2018";

Task<int> task12 = new Task<int>(new Func<int>(() => { return int.Parse(str); }));

task12.Start();

Console.Read();

}

//内部类

private class Params

{

public string name = "Rose";

public int age = 20;

public string city = "US";

}

}

}