1) Что такое TPL? Как и для чего используется тип Task

Библиотека параллельных задач TPL (Task Parallel Library) позволяет распараллелить задачи и выполнять их сразу на нескольких процессорах (для создания многопоточных приложений) Задача (task) – абстракция более высокого уровня чем поток

2) Почему эффект от распараллеливания наблюдается на большом количестве элементов?

да

3) В чем основные достоинства работы с задачами по сравнению с потокми?

►Ограничения класс Thread: ►1) отсутствует механизм продолжений ►2) затруднено получение значение результата из потока ►3) повышенный расход памяти и замедление работы приложения

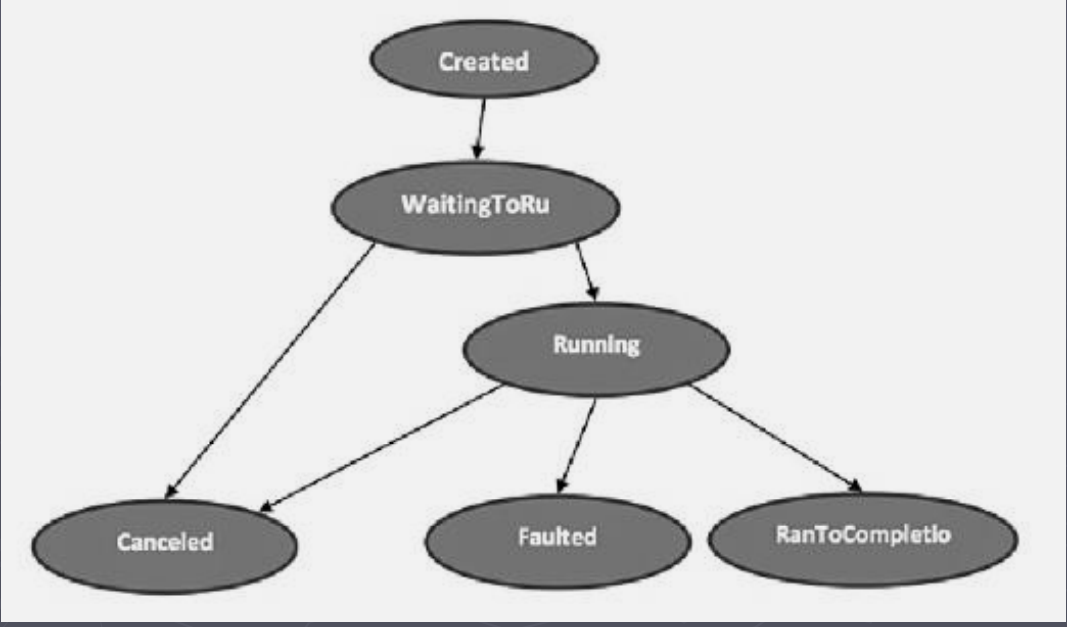
4) Приведите три способа создания и/или запуска Task?

Task task1 = new Task(() => { i++; Console.WriteLine("Task 1 finished"); }); task1.Start();

Task task2 = Task.Factory.StartNew(() => { ++i; Console.WriteLine("Task 2 finished"); });

Task task3 = Task.Run(() => { ++i; Console.WriteLine("Task 3 finished"); });

5) В каких состояниях может быть поток?Как и для чего используют методы Wait(), WaitAll() и WaitAny()?



Wait(), WaitAll() и WaitAny() останавливают основной поток до завершения задачи (или задач)

6) Приведите пример синхронного запуска Task?

task1.Start();

7) Как создать задачу с возвратом результата?

Func func = () => { Thread.Sleep(1000); return ++i; }; Task task = new Task(func); task.Start(); Console.WriteLine(task.Result); Console.WriteLine("Main finished");

8) Как обработать исключение, если оно произошло при выполнении Task?

Task task5 = Task.Run(() => {

throw new Exception(); });

try

{

task5.Wait();

}

catch (AggregateException ex)

{

var message = ex.InnerException.Message;

Console.WriteLine(message);

}

9) Что такое CancellationToken и как с его помощью отменить выполнение задач?

CancellationToken - токен отмены

10) Как организовать задачу продолжения (continuation task) ?

Task task7 = task6.ContinueWith(t => Console.Write("continuation"));

11) Как и для чего используется объект ожидания при создании задач продолжения?

Объект ожидания – это любой объект, имеющий методы OnCompleted() и GetResult() и свойство IsCompleted.

12) Поясните назначение класса System.Threading.Tasks.Parallel?

позволяет распараллеливать циклы и последовательность блоков кода

13) Приведите пример задачи с Parallel.For(int, int, Action< int >)

Parallel.For(1, 10, z=>

{ int r = 1;

for (int y = 1; y <= 10; y++)

{

r \*= z;

}

});

14) Приведите пример задачи с Parallel.ForEach

ParallelLoopResult listFact = Parallel.ForEach<int>

(new List<int>() { 1, 3, 5, 8 },

Factorial);15) Приведите пример с Parallel.Invoke()

Parallel.Invoke(

() => new

WebClient().DownloadFile("http://www.belstu.by", "start.html"),

() => new

WebClient().DownloadFile("http://www.go.by", "go.html"));

16) Как с использованием CancellationToken отменить параллельные операции?

CancellationTokenSource cancelTokenSource = new

CancellationTokenSource();

CancellationToken token = cancelTokenSource.Token;

Parallel.ForEach<int>(

new List<int>() { 1, 2, 3, 4, 5, 6, 7, 8 },

new ParallelOptions

{ CancellationToken = token},

Factorial);

}

17) Для чего используют BlockingCollection< T >, в чем ее особенность?

операции добавления и удаления блокируют доступ к коллекции для других потоков

18) Как используя async и await организовать асинхронное выполенение метода?

да