Прохождение собеседования на вакансию "Разработчик на языке C#"

Процесс отбора кандидата. Техническое собеседование.

[Процесс отбора кандидата](#_2et92p0)

[Отбор резюме](#_3dy6vkm)

[Собеседование по общим вопросам](#_4d34og8)

[Техническое задание](#_17dp8vu)

[Техническое собеседование](#_3rdcrjn)

[Итоговое собеседование](#_26in1rg)

[Возможные вопросы и задания на техническом собеседовании.](#_lnxbz9)

[Платформа .Net](#_35nkun2)

[ООП](#_44sinio)

[Язык С#](#_z337ya)

[Базы данных](#_1y810tw)

[WPF](#_2xcytpi)

[Практика](#_c5d9qfauyv0k)

[Задача 1.](#_syag5kz2wnjz)

[Задача 2.](#_gl18ip5e0474)

[Задача 3.](#_crlb0j99flzz)

[Домашнее задание](#_3whwml4)

[Дополнительные материалы](#_2bn6wsx)

[Используемая литература](#_3as4poj)

# 

# Процесс отбора кандидата

Как правило, само по себе техническое собеседование является только часть общего процесса отбора кандидата.

Я опишу вкратце наиболее распространённый вариант этого цикла, исходя из своего опыта. Количество ступеней и их периодичность может варьироваться.

Отбор включает в себя следующие этапы и их порядок:

1. Отбор резюме.
2. Собеседование по общим вопросам.
3. Техническое задание.
4. Техническое собеседование.
5. Итоговое собеседование.

## Отбор резюме

Данный процесс, собственно, сводится к тому, что HR-менеджер либо ищет интересующие резюме на сайтах вакансий, либо отбирает из уже присланных. Как правило, HR-менеджер не является ИТ-специалистом, он знает примерные требования к соискателям, какие-то технологии, которые необходимо знать и сличает с тем, что есть в резюме.

Естественно, Ваше резюме просматривает и специалист. Он, как правило, и даёт отмашку на Вашу кандидатуру. После этого, HR-менеджер назначает собеседование.

## Собеседование по общим вопросам

Происходит обычно по skype и голосом или по телефону. HR-менеджер расскажет Вам о компании, о проекте, на который необходим специалист. Разузнает о Вас, о том, зачем ищите работу, об опыте, знаниях и умениях и обязательно спросит о уровне ожидаемого дохода.

Разговор о зарплате - это отдельный разговор, выходящий за рамки данного повествования. Но Вы должны ориентироваться в предложениях по вакансиям в Вашем регионе. У компании всегда есть вилка цен, т.е. на одну и ту же должность два человека могут устроиться с разной зарплатой, это факт.

По времени может занимать до 30 минут.

## Техническое задание

В большинстве случаев Вам предложат его сделать. Необходимо это для того, чтобы первоначально оценить Ваш уровень, а код скажет о нём очень много. Если техзадание впечатлит тимлида, то на собеседовании мучать сильно не будут, возможно, даже совсем его не будет.

Оцениваться будет не красота интерфейса или крутость примененных библиотек (хотя обычно ты свободен в выборе технологий и библиотек), и это важно понимать. Будут оцениваться: структура, архитектура, чистота кода, примененные подходы к решению нетривиальных задач. Здесь хочется напомнить, что код должен следовать принципам SOLID, и Вы должны эти принципы понимать. Настоятельно рекомендуем для прочтения книгу Стива Макконелла “Совершенный код”.

По темам для технических заданий. Для джуниор-разработчика будет что-то простое, и классическим примером здесь уже стал чат или что-то с парсингом.

Для мидл-разработчика - тут выбор более широк. Как правило, будут CRUD-приложения на различные темы, хотя часто попадаются и другие задания. Например, что-нибудь на рефлексию, WCF, сетевому взаимодействию или, например, конечным автоматам. Из технологий - как WPF или ASP.Net на выбор, так и все вместе.

Так как наш курс затрагивает WPF, хочу особое внимание обратить на две важные вещи:

* Архитектура приложения соответствует MVVM и правильно применяется при этом;
* Наличие в проекте юнит-тестов и понимание, зачем они нужны.

Обычно компании не ставят каких-либо ограничений по времени выполнения задания, естественно, в рамках приличия. То есть, если Вам необходима неделя, то это не будет проблемой. Обычно, приемлемое время выполнения - 3-4 дня.

## Техническое собеседование

Если уровень выполнения технического задания окажется приемлемым, то Вас пригласят пройти техническое собеседование с вашим будущем руководителем или старшим разработчиком.

Более подробно эту тему разберём позже.

По времени может занимать до полутора часов.

## Итоговое собеседование

В случае успешного прохождения технического собеседования, далее следует итоговое, где будет сделано предложение о работе, обговаривается Ваша зарплата, испытательный срок, дата выхода на работу и другие вопросы.

# Возможные вопросы и задания на техническом собеседовании.

Состав вопросов может значительно отличаться для джуниоров и мидлов.

Для джуниор-разработчиков будут вопросы, который касаются:

* Платформа .Net;
* Язык С#;
* ООП;
* Базы данных;
* WPF.

Для мидл-разработчиков спектр вопросов будет гораздо шире, затрагивая и шаблоны проектирования и даже JavaScript и другие вопросы.

Давайте разберём наиболее часто задаваемые вопросы для джуниоров по темам.

## Платформа .Net

Помимо момента о том, что вообще собой представляет платформа, вопросы могут быть, например, следующими:

1. GC. Сборка “мусора”.
2. Что такое сериализация?
3. Что такое рефлексия?
4. Как можно изменить private поле у объекта во время runtime в обход компилятора?
5. Что такое JIT компилятор?
6. Что такое CLR?
7. Что такое сборка?
8. Разница между пространством имён и сборкой?
9. Что такое GAC?

|  |  |
| --- | --- |
| **№ вопроса** | **Краткий ответ** |
| 1 | Служба, высвобождающая память. Стек и куча. Поколения объектов. Large Object Heap. GC.Collect() |
| 2 | Сохранение состояния объекта в последовательность бит. |
| 3 | Определение типов во время выполнения |
| 4 | С помощью рефлексии |
| 5 | Компиляция кода во время исполнения |
| 6 | Common Language Runtime - общеязыковая исполняющая среда |
| 7 | Один или несколько файлов, содержащие некую функциональность |
| 8 | Пространство имен - логическое соглашение, используемое во время разработки, Сборка - область видимости имени в процессе выполнения. |
| 9 | Глобальный кэш сборок |

Для подготовки можно посоветовать - Рихтер “CLR via C#”.

## ООП

Вопросы менее привязаны к конкретному языку, но есть свои особенности:

1. Основные принципы ООП. Пример реализации каждого в C#.
2. Варианты реализации наследования.
3. Разница между интерфейсом и абстрактным классом.
4. Когда происходит вызов конструктора статического класса?

|  |  |
| --- | --- |
| **№ вопроса** | **Краткий ответ** |
| 1 | Наследование. Полиморфизм. Инкапсуляция. |
| 2 | Наследование, композиция, агрегация. |
| 3 | Абстрактный класс - наследуется,  Интерфейс - только реализуется; нельзя объявлять поля, конструктор, статические методы, у методов нет реализации. |
| 4 | Происходит при первом обращении к этому типу. |

## 

## Язык С#

1. Что такое лямбда-выражение?
2. Чем отличается явное приведение типов от неявного?
3. Какую роль играет интерфейс IDisposable?
4. Что такое ссылочные и значимые типы? В чём разница?
5. Что такое Extension методы? Для чего используются?
6. Что такое перечисления?
7. Что такое boxing и unboxing? Когда она происходит?
8. Потокобезопасные коллекции.
9. В чём преимущество параметризированных классов и методов?
10. Есть ли в C# множественное наследование?
11. От какого класса наследуются все классы в .Net.

|  |  |
| --- | --- |
| **№ вопроса** | **Краткий ответ** |
| 1 | Упрощённая запись анонимных методов. |
| 2 | Явное приведение считается небезопасным. |
| 3 | Очистка ресурсов после их использования. |
| 4 | Значимые - размещение в стеке, ссылочные в управляемой куче. Ссылочные типы существуют только в упакованной форме. Значимые типы являются производными от System.ValueType. |
| 5 | Позволяют добавлять новые методы в уже существующие типы без создания нового производного класса. |
| 6 | Набор именованных констант. Все enum являются производными от System.ValueType. |
| 7 | Упаковка/Распаковка. Для преобразования значимого типа в ссылочный служит упаковка (boxing). Для обратного преобразования - распаковка (unboxing) |
| 8 | ConcurrentQueue<T>, ConcurrentStack<T>, ConcurrentDictionary<TKey, TValue>, BlockingCollection<T> |
| 9 | Могут работать с любыми типами. Производительность у таких классов лучше, т.к. у них меньше абстрактных и виртуальных методов. |
| 10 | Нет, но возможна реализация множественных интерфейсов |
| 11 | System.Object |

## 

## Базы данных

1. Что такое T-SQL?
2. Что такое транзакция?
3. Виды изоляций транзакций?
4. Что такое хранимая процедура?
5. Что такое View?
6. Что такое индексы? Какими бывают?
7. Что такое триггеры?
8. Что такое Full Table Scan?
9. Что такое Constraint?

|  |  |
| --- | --- |
| **№ вопроса** | **Краткий ответ** |
| 1 | Процедурное расширение языка SQL |
| 2 | Последовательность SQL команд, выполняемых как единое целое. |
| 3 | * Незавершенное чтение (read uncommitted); * Подтвержденное чтение (read committed); * Повторяемое чтение (repeatable read); * Сериализуемость (serializable). |
| 4 | Именованная последовательность SQL команд. |
| 5 | Объект реляционной базы данных, который описывает виртуальную таблицу. |
| 6 | Объект базы данных, создаваемый с целью повышения производительности поиска данных. Бывает кластерный и некластерный. |
| 7 | Хранимая процедура особого типа, которую пользователь не вызывает непосредственно, исполнение её обусловлено действием по модификации данных. |
| 8 | Последовательный поиск данных, может занимать довольно долгое время влияя негативно на производительность. |
| 9 | Средство, с помощью которого можно задать правила, которым должны удовлетворять данные, для возможности записи в поле |

## 

## WPF

1. Что такое DataContext?
2. Что такое Binding?
3. Что такое INotifyPropertyChanged?
4. Виды контейнеров компоновки.
5. Что такое стиль?
6. Что такое триггер?

|  |  |
| --- | --- |
| **№ вопроса** | **Краткий ответ** |
| 1 | Контекст данных позволяет использовать привязку. |
| 2 | Взаимодействие двух объектов, при котором изменение одного влияет на значение другого. |
| 3 | Интерфейс, реализация которого позволяет свойству информировать о своём изменении. |
| 4 | Grid, StackPanel, WrapPanel, DockPanel, Canvas. |
| 5 | Набор некоторых свойств и их значений, которые потом могут применяться к элементам XAML. |
| 6 | Действия, выполняющиеся при изменении свойств некого стиля. |

Перечень используемых здесь вопросов, является далеко не полным и не исчерпывающим. Полные возможные варианты просто нет возможности рассмотреть, ввиду обширности темы.

Кроме вопросов, возможно, что на собеседовании будут задавать логические задачи или задачи на внимательность.

Например:

* В чём разница между string и String в C#?
* В чём разница между int и Int32 в C#?
* List наследуется от двух интерфейсов - IList и IEnumerable. Почему List<string>можно привести к IEnumerable<object>, а к IList<object> нельзя?(<http://professorweb.ru/my/csharp/charp_theory/level11/11_14.php>)
* Необходимо реализовать функцию, которая бы проверяла число на чётность, используя только битовые операции AND, OR, NOT
* Почему канализационные люки круглые?)))

Но по моим наблюдением такое случается довольно редко.

Гораздо чаще встречается просьба, особенно при собеседовании в офисе, написать какий-либо код. Это может быть класс, запрос к базе данных или реализация шаблона проектирования, возможно что-то другое. И здесь может наступить ступор, т.к. большинство без среды разработки и автодополнения совсем теряются. Поэтому необходимо время от времени “программировать” на обычном листе бумаги.

# Практика

## Задача 1.

Существуют следующие классы:

|  |
| --- |
| public class ClassA  {  public virtual void ClassMethod()  {  Console.Write("ClassA");  }  }  public class ClassB : ClassA  {  public override void ClassMethod()  {  Console.Write("ClassB");  }  } |

Какой вывод мы получим, выполнив следующий код?

|  |
| --- |
| ClassB object1 = new ClassA();  object1.ClassMethod();  ClassB object2 = new ClassB();  object2.ClassMethod();  ClassA object3 = new ClassB();  object3.ClassMethod(); |

## Задача 2.

Каков результат работы следующего кода?

|  |
| --- |
| bool a = true;  bool b = false;  bool c = !a || (!!a && !b) && (a || b);  Console.Write(c); |

## 

## 

## Задача 3.

Сколько раз выполнится цикл?

|  |
| --- |
| var value = null;  value = 10;  while (value > 0)  {  Console.WriteLine(a);  value--;  } |

# Домашнее задание

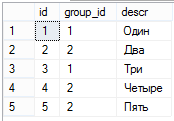
1. Выполнить без использования среды разработки, используя только ручку и лист бумаги, задачу на написание консольного приложения нахождения факториала числа N
2. Есть ли проблемы в следующем коде?

|  |
| --- |
| int i = 1;  object obj = i;  ++i;  Console.WriteLine(i);  Console.WriteLine(obj);  Console.WriteLine((short)obj); |

1. Есть таблица Users. Столбцы в ней - Id, Name. Написать SQL-запрос, который выведет имена пользователей, начинающиеся на 'A', и которые встречаются в таблице более одного раза, и их количество.
2. Какое результат вывода следующего кода?

|  |
| --- |
| private enum SomeEnum  {  First = 4,  Second,  Third = 7  }  static void Main(string[] args)  {  Console.WriteLine((int)SomeEnum.Second);  } |

1. \* Существует таблица:



Необходимо сформировать SQL-запрос, выбирающий данные следующим образом:

1. Попробовать выполнить задания, приведенные в дополнительных материалах.

# Дополнительные материалы

1. <https://tproger.ru/articles/problems/>
2. <https://habrahabr.ru/post/230881/>
3. <https://proglib.io/p/15-questions-for-programmers/>
4. <https://jopr.org/blog/detail/voprosy-na-sobesedovanii-po-c>
5. <http://surfingbird.ru/surf/dhHEA8329>

# Используемая литература

Для подготовки данного методического пособия были использованы следующие ресурсы:

1. Рихтер Дж. “CLR via C#”
2. Эндрю Троелсен - “ЯЗЫК ПРОГРАММИРОВАНИЯ C#5.0 И ПЛАТФОРМА .NET 4.5”