



# Квалификационные задания для **Ruby** разработчиков



## О заданиях

Данные задания были разработаны для облегчения проверки знаний соискателей. Задания разработаны таким образом, что вы можете их выполнить в любое удобное для вас время и в обстановке, которую считаете оптимальной.

Теперь подробнее о заданиях. Задания разделены на 3 категории.

***Level 1** позволяет оценить базовые знания, насколько вы знаете основы языка, основы программирования и насколько хорошо владеете программами, необходимыми для разработки продуктов.*

***Level 2** содержит вопросы, позволяющие оценить ваши знания по практикам разработки в Rails.*

*Ну и последняя часть – **Level 3**, в ней собраны практические задачи по разработке, в том или ином виде возникающие в процессе работы.*

Ответы на задания вы можете составить в свободной форме и прислать нам вместе с вашим резюме (не забыв указать вакансию *Ruby разработчика*) на почтовый ящик [wanted@funbox.ru](mailto:wanted@funbox.ru). После проверки заданий, мы обязательно сообщим вам о нашем решении.

В общем, это всё. Перейдя на следующую страницу, вы увидите наши вопросы и задания.



## Level I

*Здесь несколько теоретических вопросов, с помощью которых вы сможете в свободной форме рассказать нам о своем личном опыте в том или ином аспекте.*

### Q1

*Чем отличается `proc` от `lambda`?*

### Q2

*Чем отличается `&&` от `and`?*

### Q3

*Какие плюсы и минусы `Ruby` по сравнению с другими современными языками программирования вы выделяете для себя?*

### Q4

*В каком случае вы бы стали использовать `Ruby on Rails` для разработки web-приложения, а в каком — другой фреймворк?*

### Q5

*Вы запустили большой и сложный `ruby` скрипт на вашем любимом дистрибутиве `Linux`, и в процессе работы он намертво «зависает». Как найти причину сбоя?*

### Q6

*Вы запустили большой и сложный `ruby` скрипт на вашем любимом дистрибутиве `Linux`, и в процессе работы он умирает, исчерпав память, хотя, казалось бы, не должен. Как найти причину сбоя?*

### Q7

*Некоторые проекты (`Rails` и не только) мы запускаем под `JRuby`. Как вы думаете, почему?*

### Q8

*Какие плюсы и минусы библиотеки `ActiveRecord` в `Rails` вы знаете? Какие альтернативы существуют? В чем их плюсы и минусы? Какие из них вы использовали?*



## Level II

Вопросы данной категории содержат большее отношение к практике.

### Q1

Есть Rails приложение, развернутое на N серверах. В приложение нужно добавить загрузку и хранение аватаров.

1. Какую библиотеку использовать?
2. Куда складывать аватарки и почему?
3. Как и зачем нужно обрабатывать загружаемые аватарки?
4. Какие проверки загружаемого файла и когда (например, на клиенте, в Nginx или Rails) лучше всего производить?
5. Как отдавать аватарки пользователю? Какие плюсы и минусы описанного подхода?
6. Какие еще проблемы и вопросы могут возникнуть при реализации такой функциональности?

### Q2

Есть большое приложение с десятками тысяч тестов. Как лучше всего организовать код и тесты, чтобы добиться максимальной скорости прогона тестов?



## Level III

Это решающий раунд, где нужно сразиться с реальными задачами. Вы можете искать решения в интернете, гуглить, читать википедию и так далее, но помните, что в будущем вам придется постоянно решать подобные задачи.

### Q1

Напишите скрипт `nmax`, который делает следующее:

- читает из входящего потока текстовые данные;
- по завершении ввода выводит `n` самых больших целых чисел, встретившихся в полученных текстовых данных.

Дополнительные указания:

- числом считается любая непрерывная последовательность цифр в тексте;
- известно, что чисел длиннее 1000 цифр во входных данных нет;
- число `n` должно быть единственным параметром скрипта;
- код должен быть покрыт тестами;
- код должен быть оформлен в виде гема (содержащего исполняемый файл, код модулей и т.д.);
- плюсом является размещение на Github и интеграция с Travis CI.

Пример запуска:

```
cat sample_data_40GB.txt | nmax 10000
```

### Q2

Реализуйте web-приложение (Rails проект), которое удовлетворяет нижеизложенным требованиям.

- Приложение содержит две страницы: `/` и `/admin`
- На странице `/` отображается текущий курс доллара к рублю, известный приложению.
- Приложение фоновым скриптом периодически обновляет курс из любого выбранного вами доступного источника (сайт CBR, главная страница <http://www.rbc.ru>, и т.д.).
- При обновлении курса в приложении он обновляется на всех открытых в текущий момент страницах `/`.
- На странице `/admin` находится форма, содержащая поле для ввода числа, поле для ввода даты-времени и сабмит.
- При сабмите введенное число делается форсированным курсом до введенного времени, т.е. до этого времени реальный курс игнорируется, вместо него страницах `/` отображается форсированный курс.



- Страница /admin «помнит» введенные предыдущий раз значения, они отображаются уже введенными при загрузке страницы.
- При сбросе форсированного курса он, конечно же, сразу обновляется на всех открытых страницах /. При истечении времени действия форсированного курса на всех страницах начинает отображаться реальный курс.
- Форма содержит разумные валидации.
- Внешний вид приложения должен быть аккуратным в рамках разумного (например, использовать Twitter Bootstrap).
- Плюсом будет использование какого-либо JS-фреймворка на клиентской стороне.
- Web-приложение должно корректно работать в браузерах Firefox и Chrome последних версий.
- Код должен быть покрыт тестами.
- Все необходимое для локального запуска приложения должно быть оформлено в виде Procfile-а для Foreman.

*Спасибо за время, потраченное на выполнение заданий!*