Danh sách câu hỏi phỏng vấn DTL

<HienPT13>

1. Tên em là gì ? sinh năm bao nhiêu ? Học trường nào?
2. Giới thiệu bản thân
3. Nêu các tính chất SOLID

* Single reposibility principle: Mỗi lớp chỉ chịu trách nhiệm về một nhiệm vụ cụ thể nào đó thôi. Ví dụ nếu làm về quản lí sản phẩm thì nên làm thêm, sửa, xóa còn In, xuất file thì nên để ở một class khác làm. Hay ghi log lỗi thì nên tách ra một lớp khác làm. (Liên quan đến tính đóng gói trong OOP)
* Open-Closed principle: Chúng ta nên hạn chế chỉnh sửa bên trong một module, class có sẵn để tránh ảnh hướng đến tính đúng đắn của chương trình. Khi cần thêm tính nắng mới thì ta nên kế thừa hoặc mở rộng các class, module có sẵn thành các module con lớn hơn. Các module/class con vừa có đặc tính của lớp cha ( đã kiểm tra tính đúng đắn từ trước) vừa có đặc tính mởi vừa được bổ sung phù hợp với yêu cầu. (Liên quan đến tính kế thừa trong OOP)
* Liskov substitution principle: Các instance của lớp con có thể thay thể các instance của lớp cha mà vẫn đảm bảo tính đúng đắn của chương trình. (Liên quan đến tính đa hình trong OOP)
* Interface segregation principle: Nếu interface quá lớn thì nên tách thành các interface nhỏ hơn với nhiều mục đích cụ thể. Thay vì có hang tram method trong 1 interface thì ta nên chia thành nhiều nhóm nhỏ. Để khi có một lớp implement thì không cần thực hiện hết tất cả những method không cần thiết.
* Dependency inversion principle: Các module cấp cao không nên phụ thuộc vào các modules cấp thấp mà phụ thuộc vào abstraction, các class giao tiếp với nhau thông qua interface không phải thông qua implementation.

1. Lấy ví dụ về áp dụng Single
2. Lấy ví dụ về open/close mà em từng áp dụng vào trong project
3. Lấy ví dụ về Interface subtiation principle
4. Áp dụng depency injection trong project ở fresher như thế nào
5. Ứng dụng của solid trong project
6. Mock project em làm về gì?
7. Nhiệm vụ của em làm gì trong mock project đó
8. Em dùng kỹ thuật nào để phân trang? Em phân trang ntn
9. Trong mock project em có dùng Entity framework không?
10. Ưu nhược điểm của EF

O/RM (Object Reloational mapping) là kỹ thuật chuyển đổi dữ liệu giữa các hệ thống không tương thức, giữa RDBMS và OOP. ORM tự động chuyển đổi từ object sang table hay table sang object.

Entity Framework là một ORM Framework trong .NET framework. Là một kỹ thuật truy cập CSDL giúp lập trình viên thao tác vào CSDL một cách nhanh chóng.

Nhược:

* Chậm.
* Phức tạp.

Ưu điểm:

* Code ngắn gọn, tương tự với code linq trong c#
* Cung cấp tự động tạo code.
* Hỗ trợ code first, cho phép thiết kế database dựa trên các class.
* Có thể làm việc trên nhiều CSDL khác nhau: Oracle, DB2, MySql, SqlServer… không phụ thuộc vào mô hình lưu trữ.

1. Trong mock em dùng ajax cho những chức năng nào?
2. Dùng ajax thì gọi hàm nào

Một số phương thức của Ajax là $.get(), $.post(), $.load(), $.ajax(). 3 hàm get, post, load có thể được coi là một ajax

1. Cơ chế truyền ajax
2. Em có dùng Jquery nhiều không?
3. Sử dụng index trong SQL
4. Ưu nhược điểm dùng index .   
   Ưu điểm : nhanh, Nhược điểm là làm chậm câu truy vấn insert,update,delete.
5. Giả sử có nhiều CSDL từ nhiều năm trước, nêu cách thiết kê DB để dễ dàng tìm kiếm và truy xuất
6. Ưu nhược điểm của store procedure

Nhược:

Tốn tài nguyên trên server

Khó viết các SP thực hiện các nghiệp vụ phức tạp.

Khó debug

Khó bảo trì.

1. So sánh store và câu lệnh sql
2. 4 bước thực hiện store procedure là gì?

B1. Check syntax

B2. Complied.

B3. Run.

B4. Return value.

1. Cách tăng performent store procedure
2. Em có dùng …type trong store procedure không?
3. Em có được học về Cấu trúc dữ liệu và giải thuật.
4. Nêu về đệ quy

Là chương trình có thể gọi lại chính nó

Khi chương tình gọi lại chính nó, mục đích giải quyết là giải quyết vấn đề tương tự nhưng nhỏ hơn

Vấn đề nhỏ hơn này cho tới một lúc nào đó sẽ đơn giản đến mức chương trình có thể tự giải quyết được mà không cần gọi lại nữa gòi là Điều kiện dừng.

1. Ví dụ về đệ quy

Tìm UCLN của 2 số

|  |  |
| --- | --- |
| 3  4  5  6  7  8  9 | int UCLN(int a,int  b)  {  if (b == 0)  return a;  else  {  return UCLN(a, a%b);  }  } |

1. Chỉ dùng queue, nếu cách cài stack trong queue

Queue: Vào trước ra trước.Thêm mới dung Enqueue, Xóa phần tử dung Dequeue.

Stack: Vào trước ra sau: Push để thêm mới. Pop để lấy ra.

1. Nêu ví dụ về stack và queue.
2. Phân biệt string và Stringbuilder

|  |  |
| --- | --- |
| String | String Builder |
| Dạng Immutable không thể thay đổi | Dạng Mutable có thể thay đổi |
| Ở namespace System | Ở Namspace System.Text |
| Khi tạo ra một đối tượng thì không thể modified | Khi tạo ra một đối tượng chúng ta có thể thêm, sửa , xóa đối tượng này. |
| Không thể sử dụng append | Có thể append |
| Chạm hơn vì mỗi một thao tác lại tạo ra một đói tượng mới. | Nhanh hơn bởi vì không cần tạo ra nhiều đói tượng mới cho các thao tác thực hiện. |
| Lưu trữ trong String pool | Lưu trữ trong Heap |
|  |  |

1. Em muốn offer lương bao nhiêu?
2. Nếu phải đi làm ngay em có đi làm được không ?
3. Em có câu hỏi nào cho anh không?
4. Kể tên các thuật toán tìm kiếm mà e biết.

Linear search

Binary Search

* Jump Search

Yêu cầu danh sách sắp xếp theo thư tự.

+ Bước nhảy: Sqrt(số phần tử)

+ Sử dụng biến prev để lưu vị trí ban đầu, jump để làm biến nhảy chạy dọc các vị trí của mảng.

Bước 1: Gán Prev= 0; jump= bước nhẩy

Bước 2: So sánh jump với X.

2.1 Nếu nhỏ hơn X thì Prev=jump; jump=jump+ bước nhẩy. Quay lại bước 2.

2.2 Nếu bằng X thì trả lại kết quả. Chuyển sang bước 3.

2.3 Nếu lớn hơn X. Thì duyệt lần lược từ Prev đến jump để tìm. Chuyển sang bước 3.

Bước 3: Trả lại kết quả.

+ Độ phức tạp Sqrt(n)

* Interpolation Search (tìm kiếm nội suy):

+ Dãy số cũng phải được sắp xếp:

+ Bước 1: Từ công thức tìm post của Binary Search:

Pos= **left + (X- T[left]) \* (right – left) / (T[right] – T[left])**

T là mảng giá trị, X là giá trị cần tìm. Left, right là giá trị đầu cuối.

+ Bước 2: So sánh x với T[pos]

Nếu nhỏ hơn X thì tang left lên 1 đơn vị, thực hiện lại bước 1

Nếu lớn hơn X ta giảm right đi 1 và tiếp tục thực hiện bước 1.

+Độ phức tạp Log2(log2(n))

int InterPolationSearch(int arr[], int n, int x)

{

  int left = 0;

  int right = n-1;

  while (left <= right && x >= arr[left] && x <= arr[right])

  {

    double val1 = (double) (x - arr[left]) / (arr[right]-arr[left]);

    int val2 = (right-left);

    int pos = left + val1\*val2;

    if (arr[pos] == x)

      return pos;

    if (arr[pos] < x)

      left = pos + 1;

    else

      right = pos - 1;

  }

  return -1;

}

1. Theo em thuật toán nào là tối ưu nhất.
2. Độ phức tạp của thuật toán.
3. Khi nào dùng Quick sort,….
4. Em hay dùng design pattern nào..Có mấy loại design pattern. Nếu em biết các loại thì các loại đấy được sử dụng khi nào.Trong reponsitory tầng xử lý nghiệp vụ em hay làm gì ? dependency injection e áp dụng nó như thế nào.. Để làm gì…

Design Patterns là ác giải pháp đã được tối ưu hoa, được tái sử dụng cho các vấn đề lập trình mà chúng ta gặp phải hang ngày. Nó là một khuôn mẫu đã được suy nghĩa, giải quyết trong tình huống cụ thể rồi. Các tính vấn đề

Sử dụng Design pattern giúp tái sử dụng mã lệnh, mở rộng dễ dàng. Cung cấp giải phpas ở dạng tổng quát giúp tang tốc độ phát triển phần mềm bằng các đưa ra các mô hình test, mô hình phát triển đã qua kiểm nghiệm. Tránh được các lỗi tiềm ẩn có thể gây ra những lỗi lớn, dễ dàng nâng cấp bảo trì về sau> Giúp cho lập trình viên hiểu code của người khác 1 cách nhnah chóng.

Có 3 nhóm pattern chính: 32 mẫu

Creational Pattern (Nhóm khởi tạo 9 mẫu): Abstract Factory, Factory Method, SingleTon, Prototype: Giúp bạn trong việc khởi tạo các đối tượng. Sử dụng một số thủ thuật để không nhìn thấy từ khoa new khi cấp phát.

Structural Pattern (Nhóm cấu trúc 11 mẫu): Adapter, Bridge, Composite, Decorator, Façade Proxy, Flyweight: Dùng để thiết lập, định nghĩa quan hệ giữa các đối tượng.

Behavioral Pattern(Nhóm hành vi 12 mẫu): Interpreter, Template mothod, chain ò responsibility, command, Iterator, Mediator, state, strategry, visitor: Nhóm này dung trong thực hiện các hành vi của đối tượng.

Repository là một mẫu gần đây xuất hiện.

1. Em dùng design pattern ở trong dự án nào. (Vì nói là làm app chat trong khi chờ phỏng vấn nên anh ấy hỏi tiếp: làm thế nào để 2 người có thể tương tác trực tiếp với nhau?).
2. Em có dùng bootstrap không ? Em dùng nó như thế nào.Bây giờ cho em 1 file psd e có cắt được không.. Em cắt nó như thế nào.Theo em khó nhất khi cắt file psd là gì. Bây giờ a muốn cho các div dùng chung 1 kiểu style ví dụ như font-size,… mà trong đó có 1 div a muốn đặt kiểu style khác thì làm thế nào.
3. Em có hay dùng css không..
4. Nhược điểm của ajax.

Ưu điểm:

Giúp thiết kế web đa dạng hơn, tang tính tương tác cùa website với người dung

Nó sử dụng các công nghệ có sẵn nên dễ học và dễ sử dụng

Được hỗ trợ trong các trình duyệt phổ biến hiện nay.

Nhược điểm:

Không thể Bookmark nó vào Favourite trên trình duyệt, hay gửi link đến bạn bè.

Lỗi bảo mật do dựa trên jquery, Khi tắt jquery trên trình duyệt thì không chạy được

1. Ưu nhược điểm của Entity Framework .
2. Em dùng gì để phân trang. Em có dùng store procedure để phân trang không. Bây giờ trong câu truy vấn sql a muốn bắt đầu từ 1 bản ghi và kết thúc ở 1 bản ghi thì làm thế nào.
3. Đệ quy là gì. Em có hay dùng đệ quy trong mock không. Ngoài áp dụng vào khi áp dụng vào các thuật toán sắp xếp thì  
   bình thường em có áp dụng đệ quy trong bài làm không.
4. Ngoài Entity framework thì em có hay dùng cái gì nữa không.
5. À quên em nêu về lập trình hướng đối tượng. Các tính chất của lập trình hướng đối tượng.Ví dụ trong mock em làm. Không lấy ở ngoài.
6. Em biết csdl nào khác ngoài sql không. Em biết mysql à,…
7. Em có chơi thể thao không.Uống được rượu không.Đi tiếp khách e uống dc không. Em đi làm ngay được không…