Liberating the Programmer with Prorogued Programming

Mehrdad Afshari, Earl T. Barr, Zhendong Su

University of California, Davis
Motivation

- Mainstream PLs force abstraction shifts
Motivation

• Mainstream PLs force abstraction shifts

```csharp
static void Main() {
    Console.WriteLine("Email address: ");
    var emailAddress = Console.ReadLine();
    // Check validity of the email address
```
Motivation

- Mainstream PLs force **abstraction shifts**

```csharp
static void Main() {
    Console.WriteLine("Email address: ");
    var emailAddress = Console.ReadLine();
    // Check validity of the email address

    static bool IsValidEmail(string email) {
        throw new NotImplementedException();
    }
}
```
Motivation

• Mainstream PLs force **abstraction shifts**

```csharp
static void Main() {
    Console.WriteLine("Email address: ");
    var emailAddress = Console.ReadLine();
    // Check validity of the email address
    if (!IsValidEmail(emailAddress)) {
        Console.WriteLine("Invalid email");
    } else {
        // Process "emailAddress"
    }
}

static bool IsValidEmail(string email) {
    throw new NotImplementedException();
}
```
Motivation

• Mainstream PLs force abstraction shifts
• Shifting abstraction is distracting
Motivation

• Mainstream PLs force abstraction shifts
• Shifting abstraction is distracting
• Distraction hinders productivity
“...a programming language should, above all, be malleable. A programming language is for thinking of programs, not for expressing programs you've already thought of”

—Paul Graham
Goal

- Reduce abstraction shifts
- Align PLs with thought process
- Aid top-down design
- Promote iterative refinement
- Facilitate natural workflow
A new programming paradigm closely aligned with a programmer’s thought process by providing the ability to prorogue, defer a concern, run and experiment with incomplete code, and gradually and iteratively reify the missing parts.
prorogue |p(r)əˈrōg|
verb ( prorogues, proroguing, prorogued ) [ with obj. ]
1. defer, postpone
2. to terminate a session of (as a British parliament) by royal prerogative
static void Main() {
    Console.WriteLine("Email address: ");
    var emailAddress = Console.ReadLine();
    // Check validity of the email address
    bool isValid = prorogue IsValidEmail(emailAddress);
    if (isValid) {
        Console.WriteLine("Invalid email");
    } else {
        // Process "emailAddress"
    }
}
static bool IsValidEmail(string s) {
    if (s == "good@email.com") {
        return true;
    }
    if (s == "notvalid") {
        return false;
    }
    // fallback prorogue call:
    return prorogue;
}
double GetTemperature(string apiKey, string zipCode) {
    var client = new TemperatureService { ApiKey = apiKey };
    return client.GetTemperature(unit: “celsius”); // prorogued call
}
Principles

- Proroguing concerns
- Hybrid computation
- Executable refinement
Applications
void ProcessPaymentAndRedirect(CreditCard cc, decimal amount) {
    // temporarily prorogue the call for debugging:
    if (prorogue cc.TryCharge(amount)) {
        var videoid = (int)Session["RequestedVideo"];
        var userAddress = Request.UserHostAddress;
        var contentServer = FindClosestServer(userAddress, videoid);
        var authKey = contentServer.GenerateAuthKey(videoid);
        var url = GetVideoUrl(contentServer.Address, videoid, authKey);
        Response.Redirect(url);
    } else {
        Response.Redirect("/FailedPayment");
    }
}
Applications

- Mocking external resources
- Automatic unit test generation
- Evolving API
- Crowdsourcing
Thanks!

http://mehrdad.me/prorogue