Introduction au développement iOS

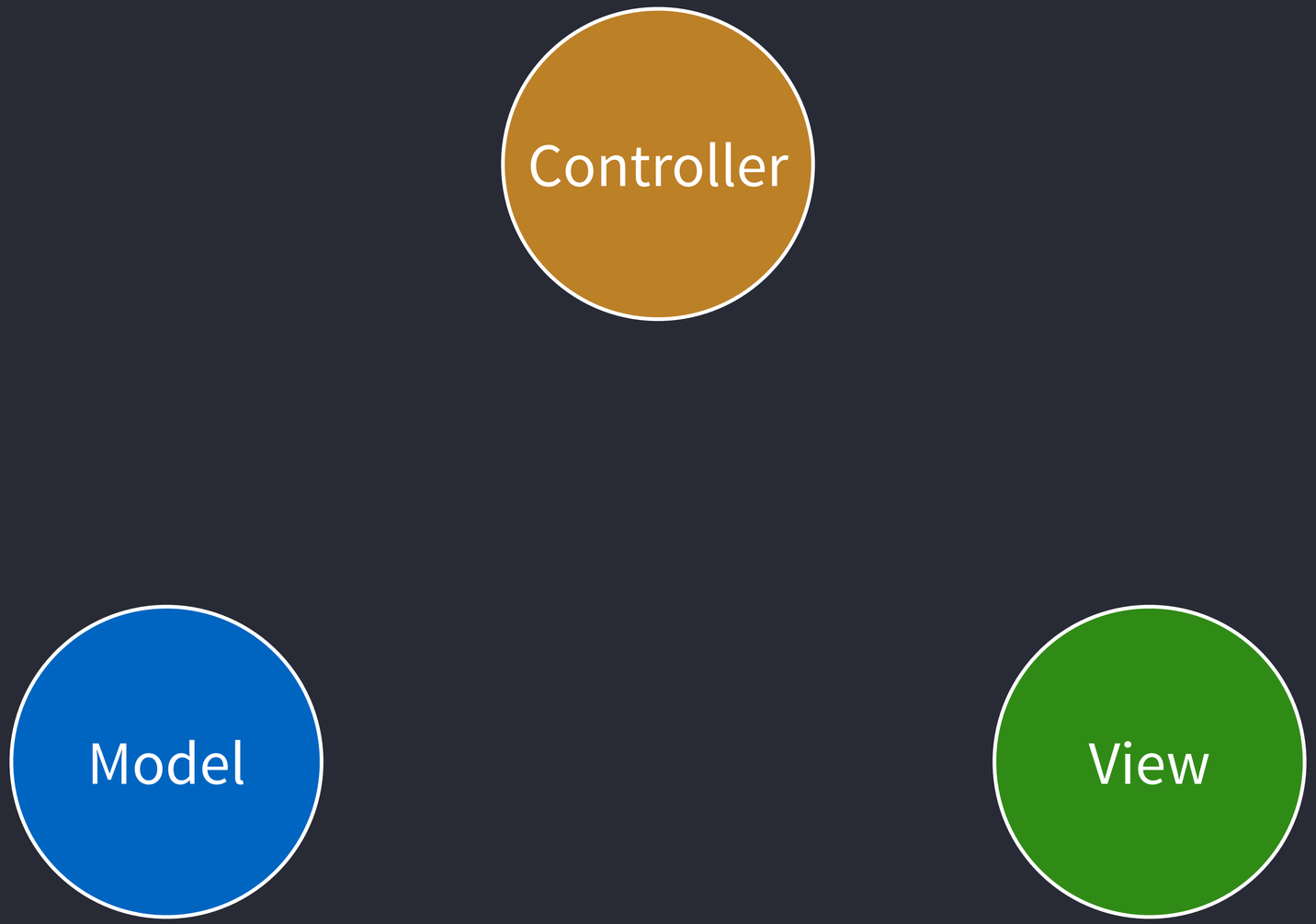
Cours 2

Adrien Humilière
adrien.humiliere@djit.fr

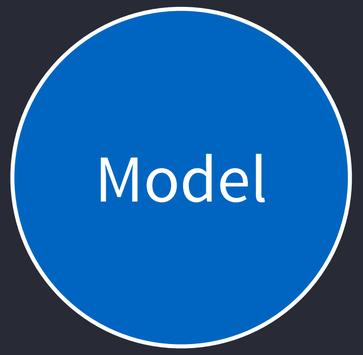
Architecture des applications

- Plusieurs types d’architectures existent.
- Le pattern MVC (Model, View, Controller) est la structure la plus répandue pour les applications iOS.

Model - View - Controller



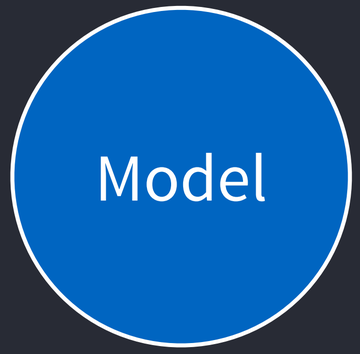
Model - View - Controller



Le coeur de l’application

Model - View - Controller

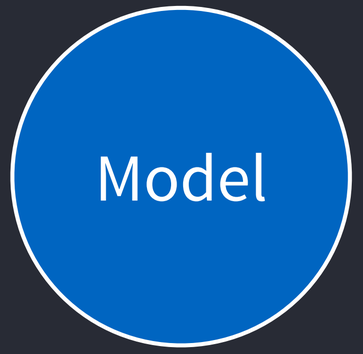
Le gestionnaire d’affichage



Le coeur de l’application

Model - View - Controller

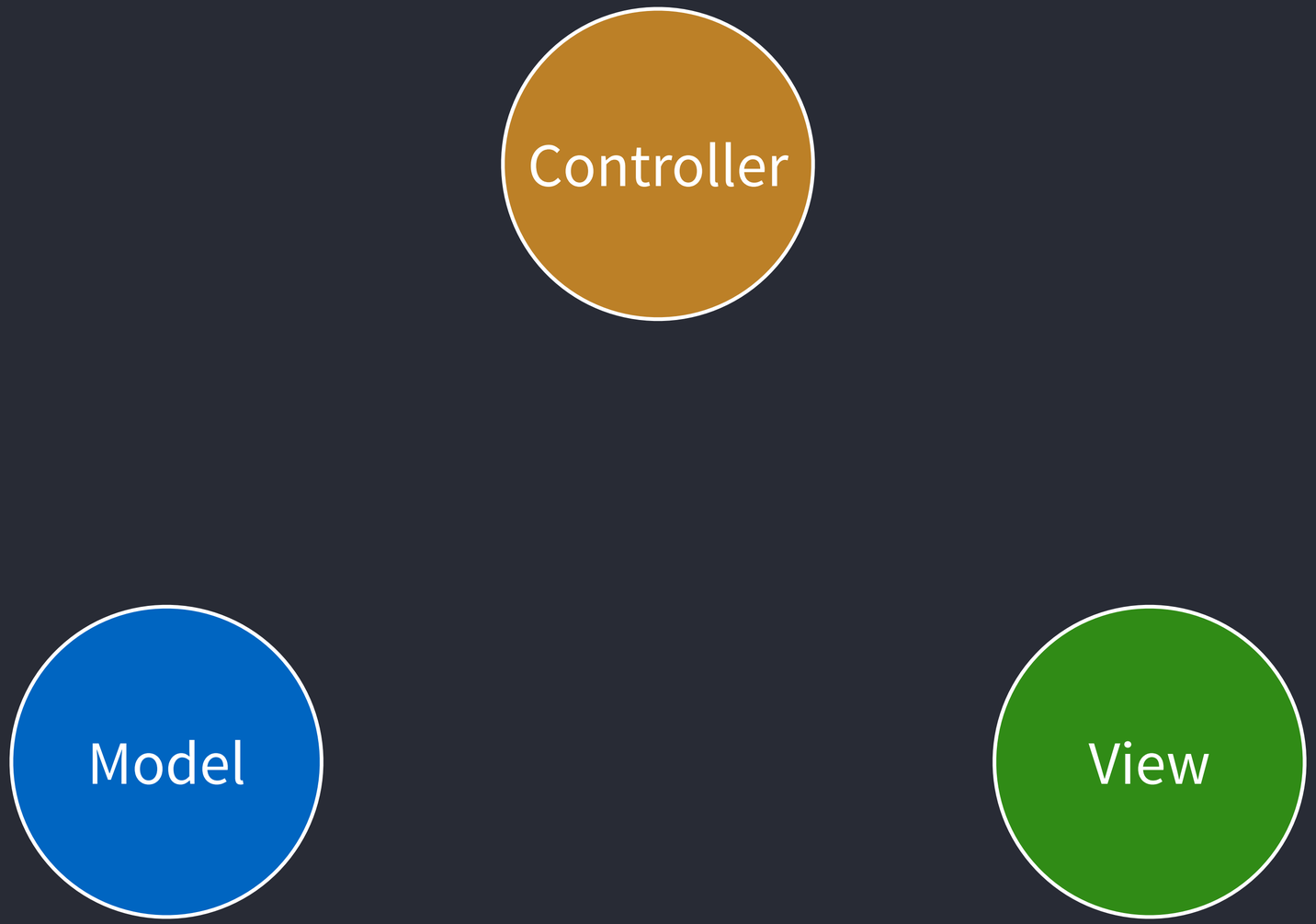
Le gestionnaire d’affichage



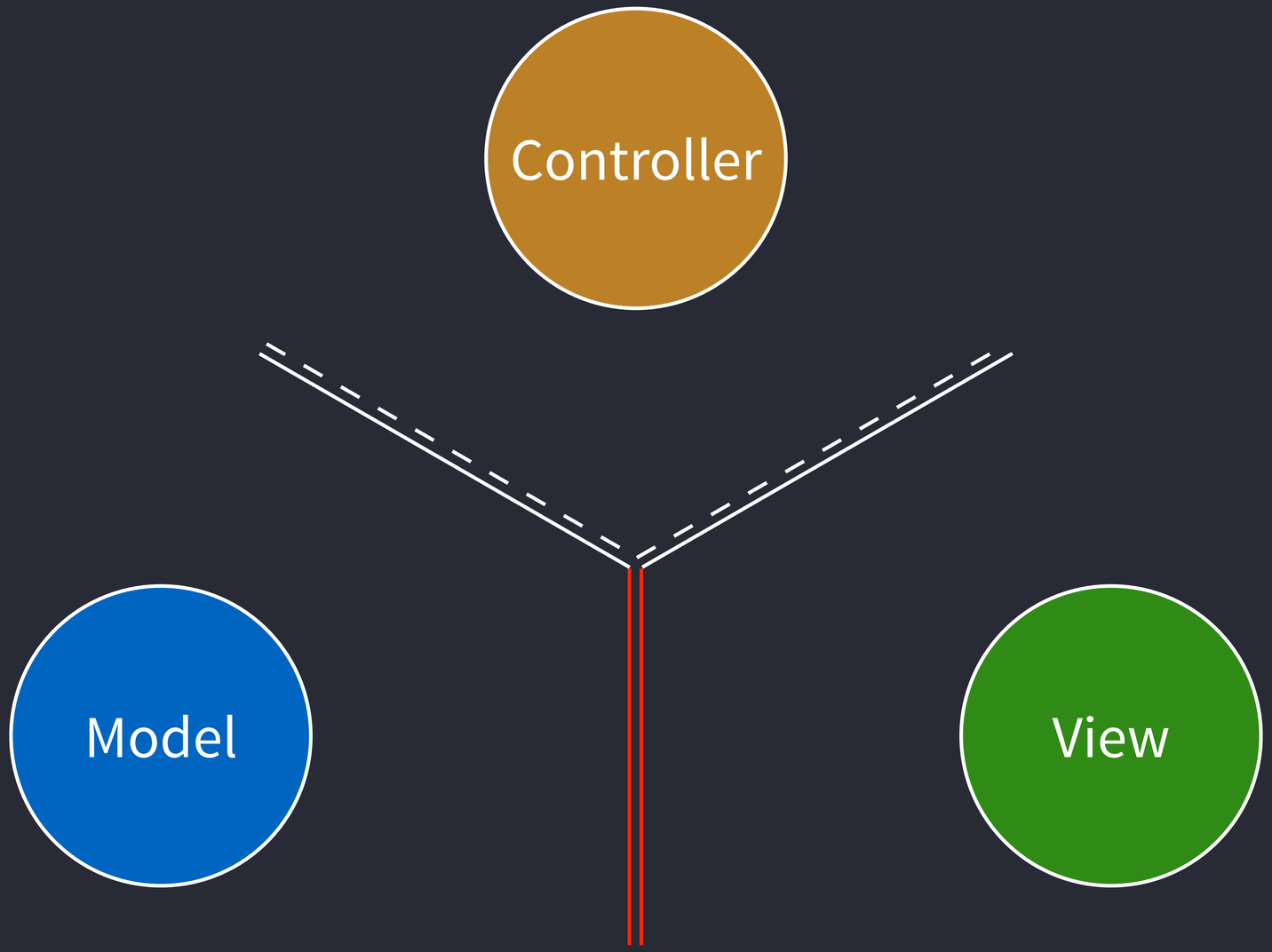
Le coeur de l’application

L’interface avec l’utilisateur

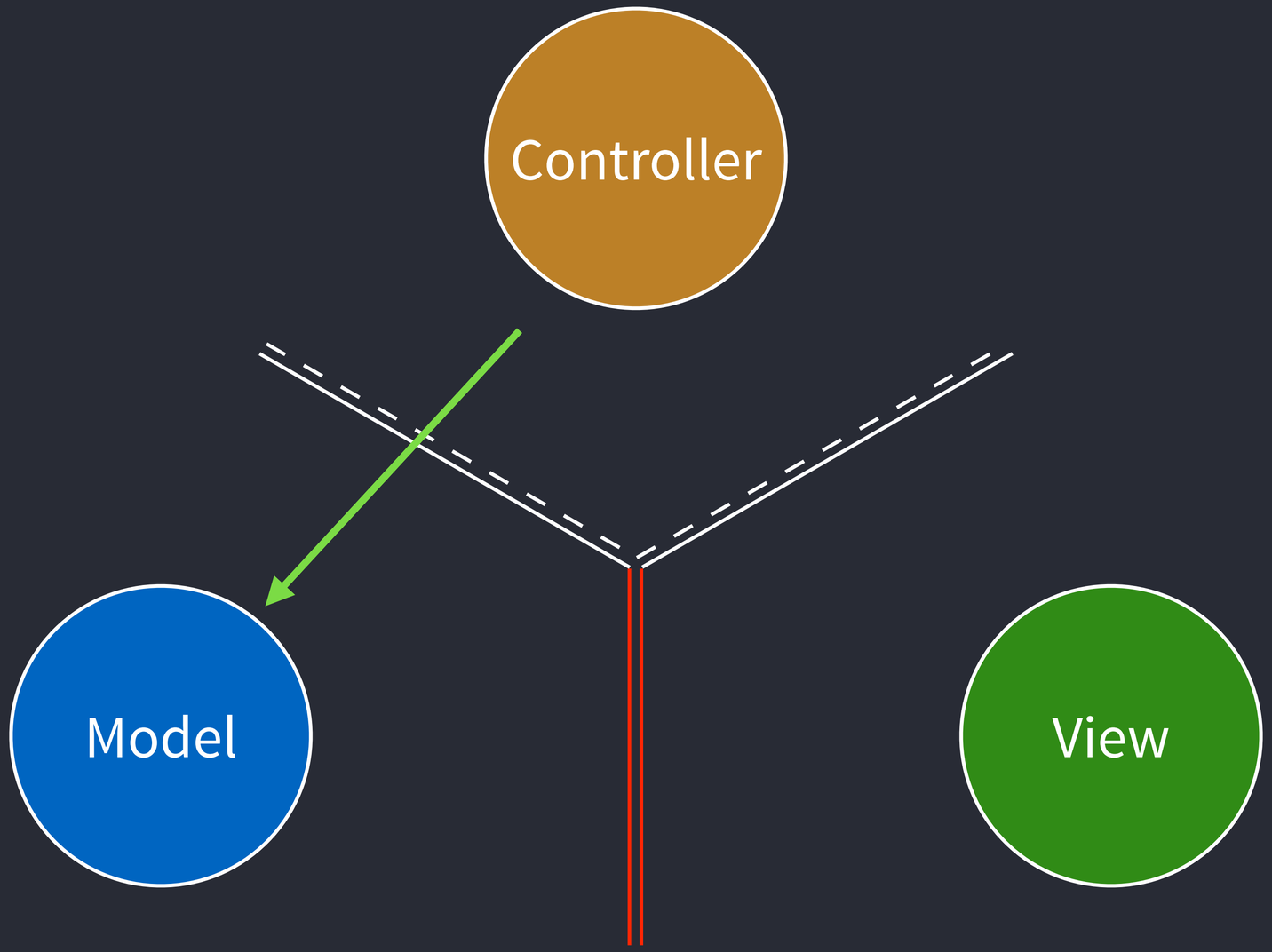
Model - View - Controller



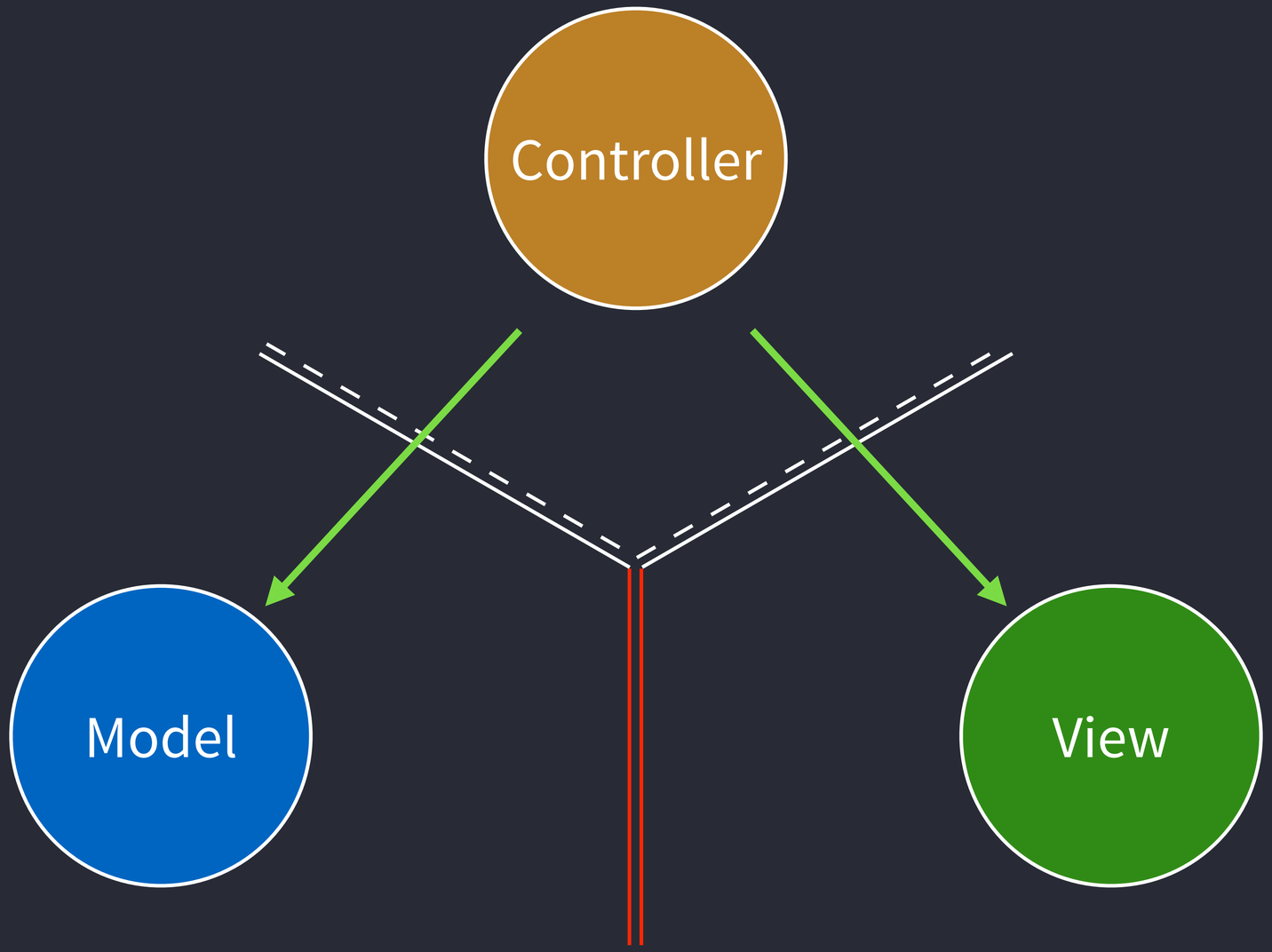
Model - View - Controller



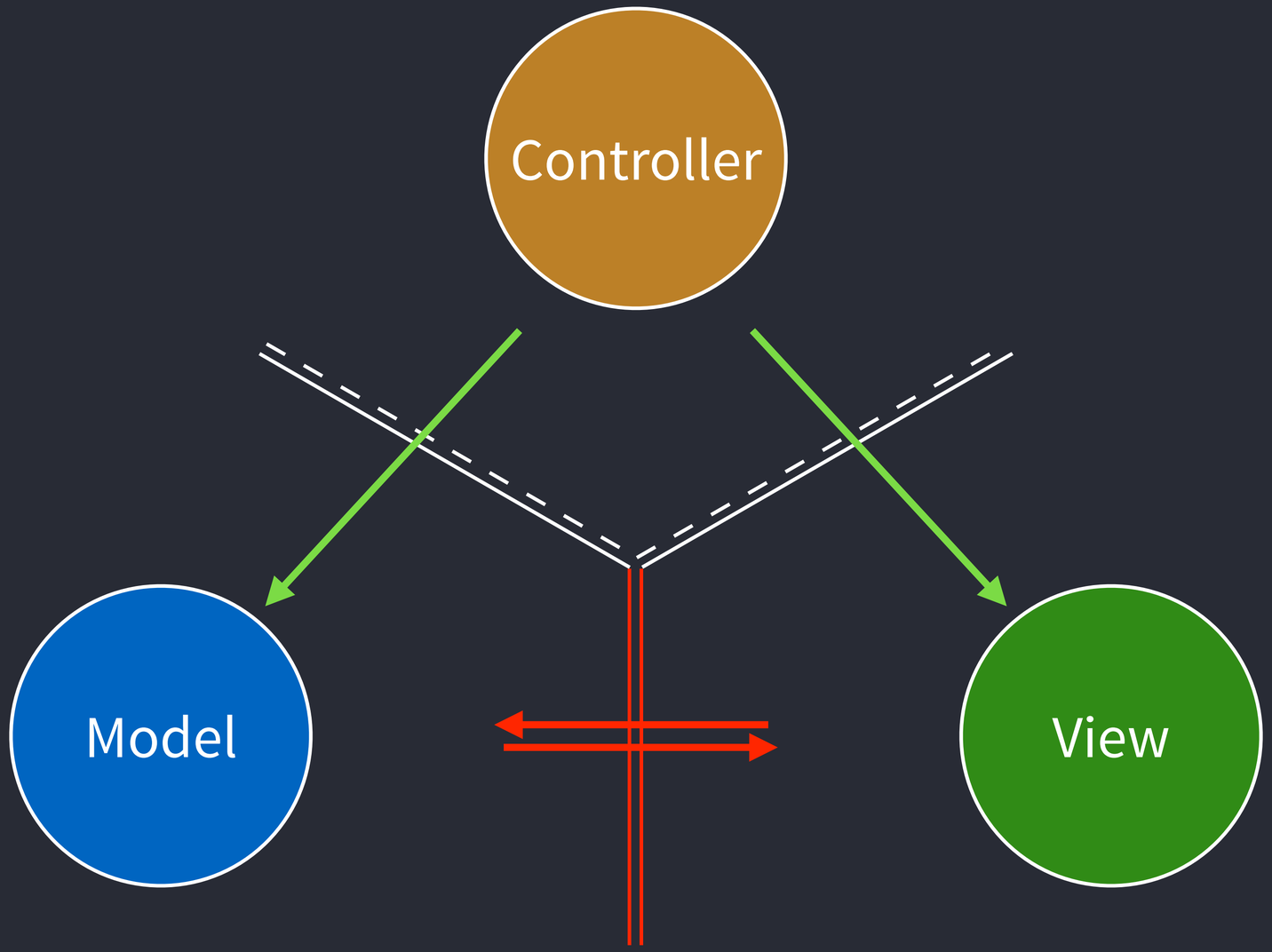
Model - View - Controller



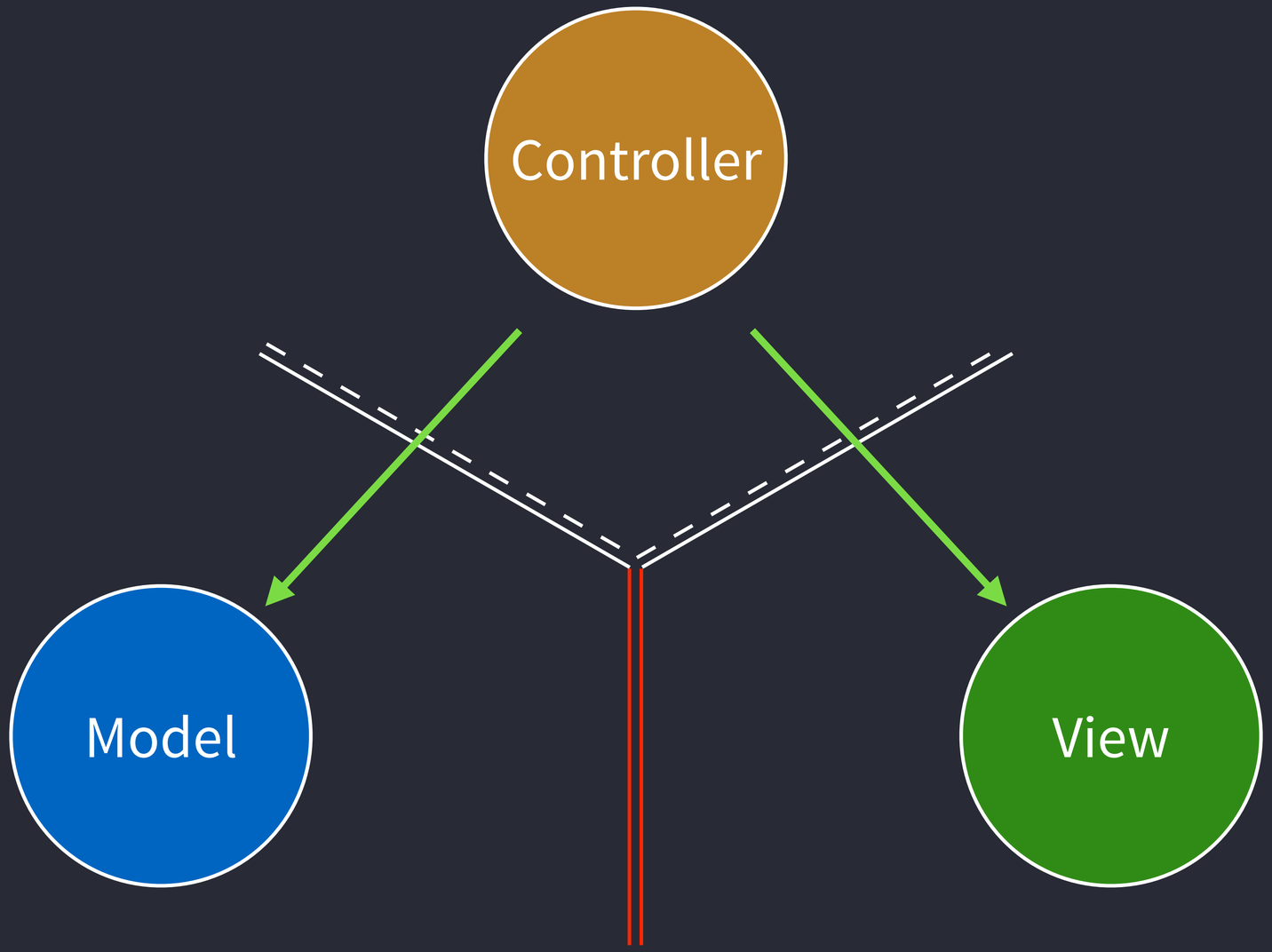
Model - View - Controller



Model - View - Controller



Model - View - Controller

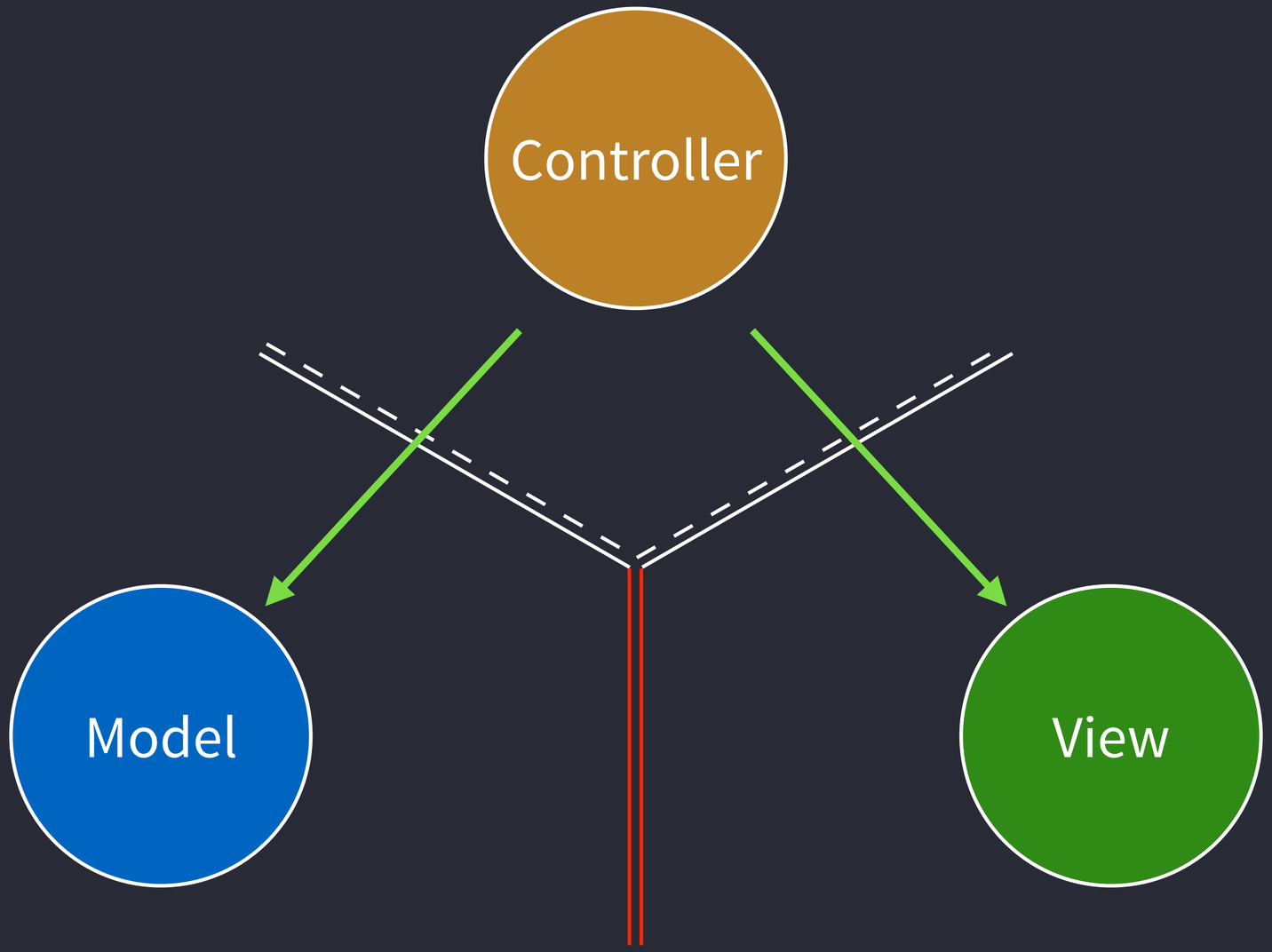


Model - View - Controller

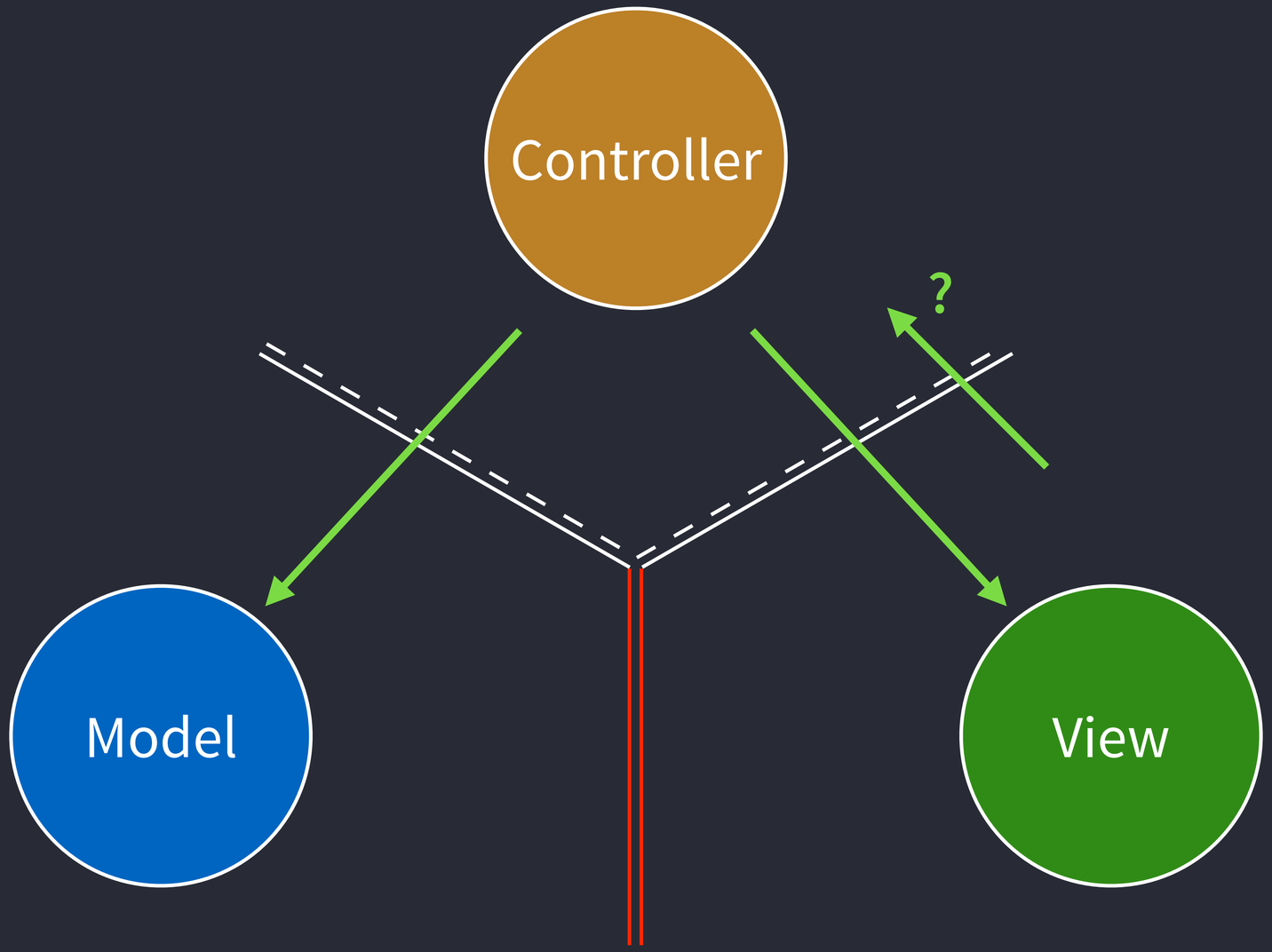
Contrôle de la view par le controller

```
- (void)viewDidLoad {
    [self.view displayLabel];
    [self.view updateWithFormattedText:self.track.title];
}
```

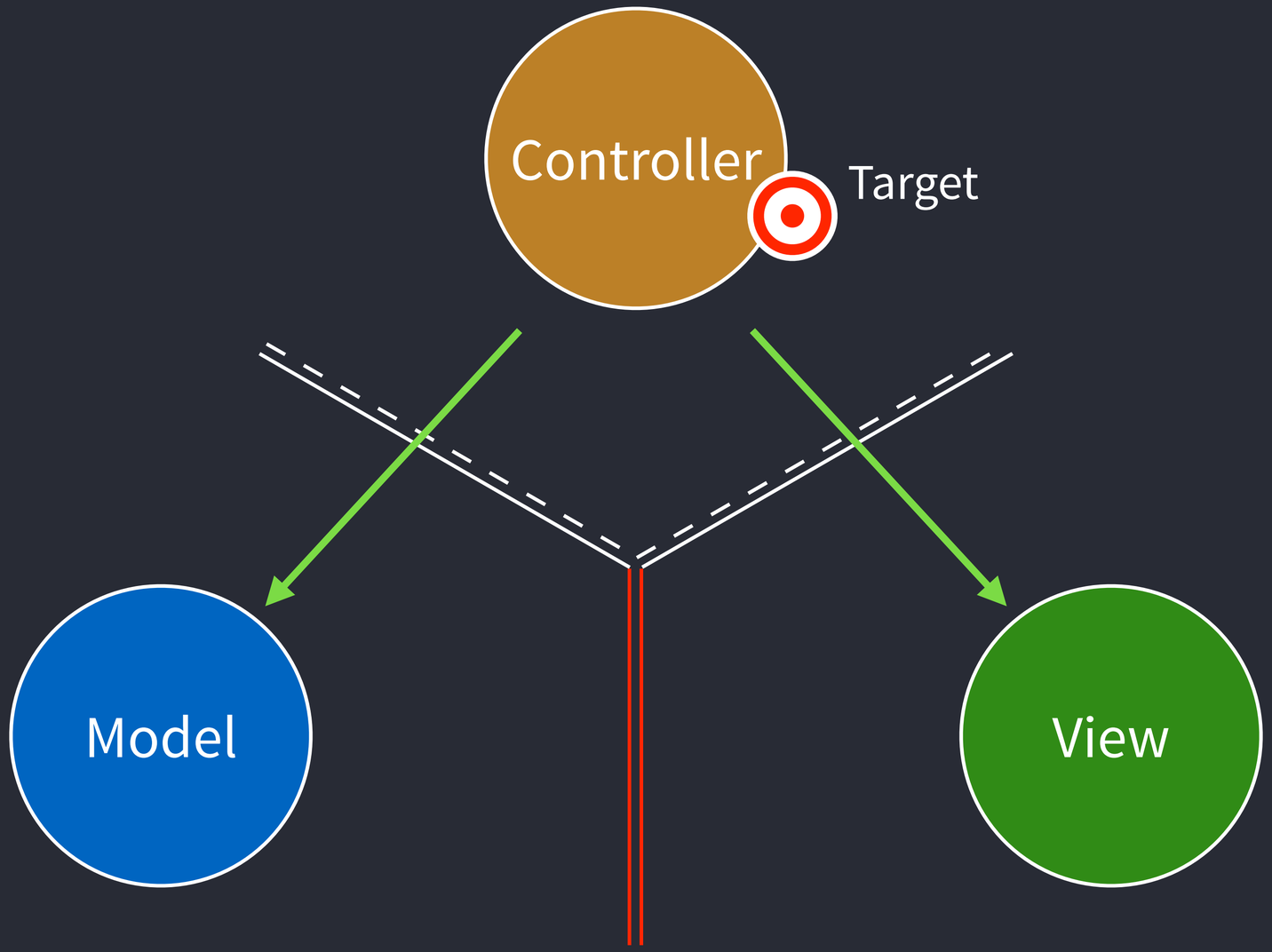
Model - View - Controller



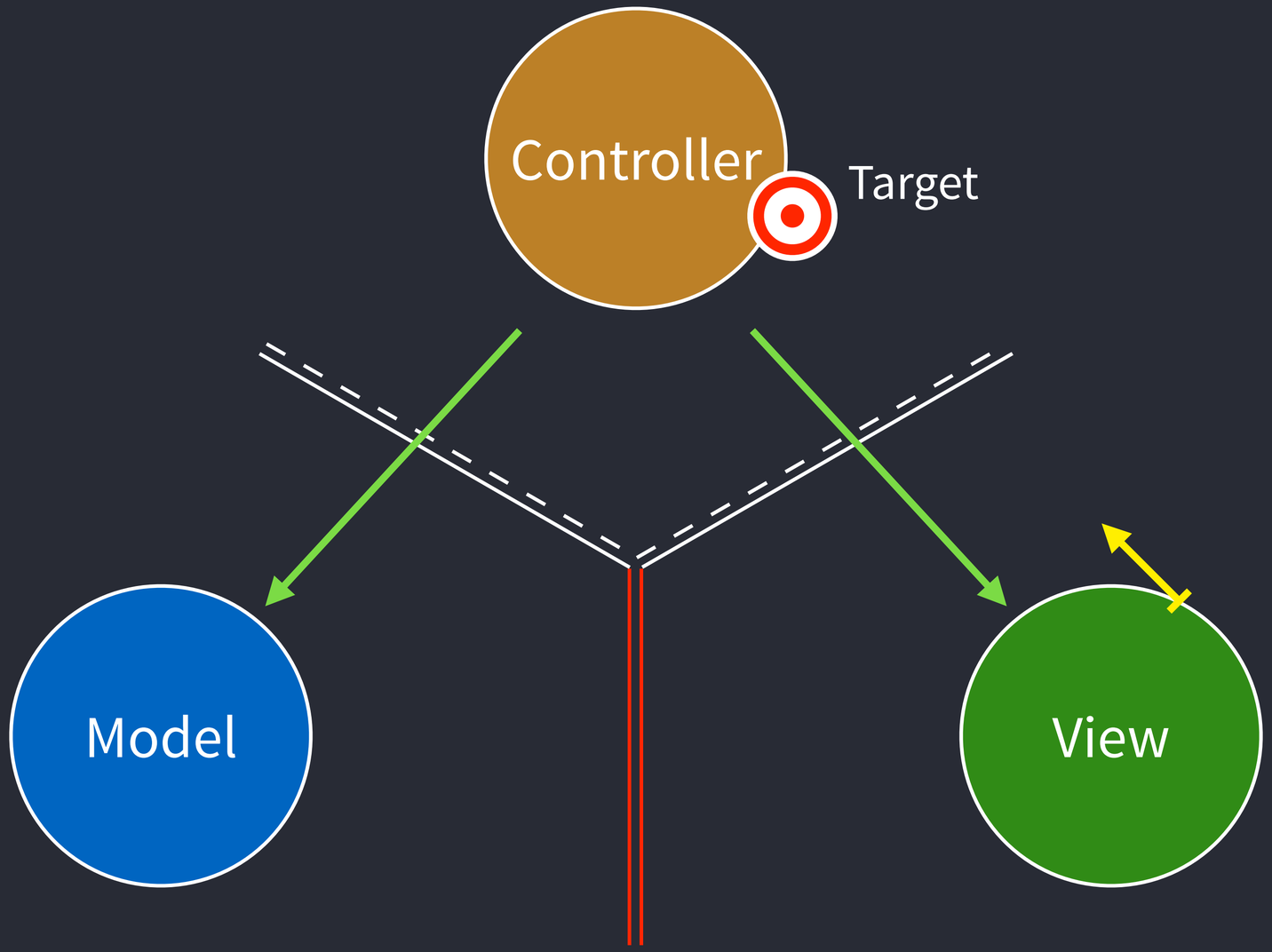
Model - View - Controller



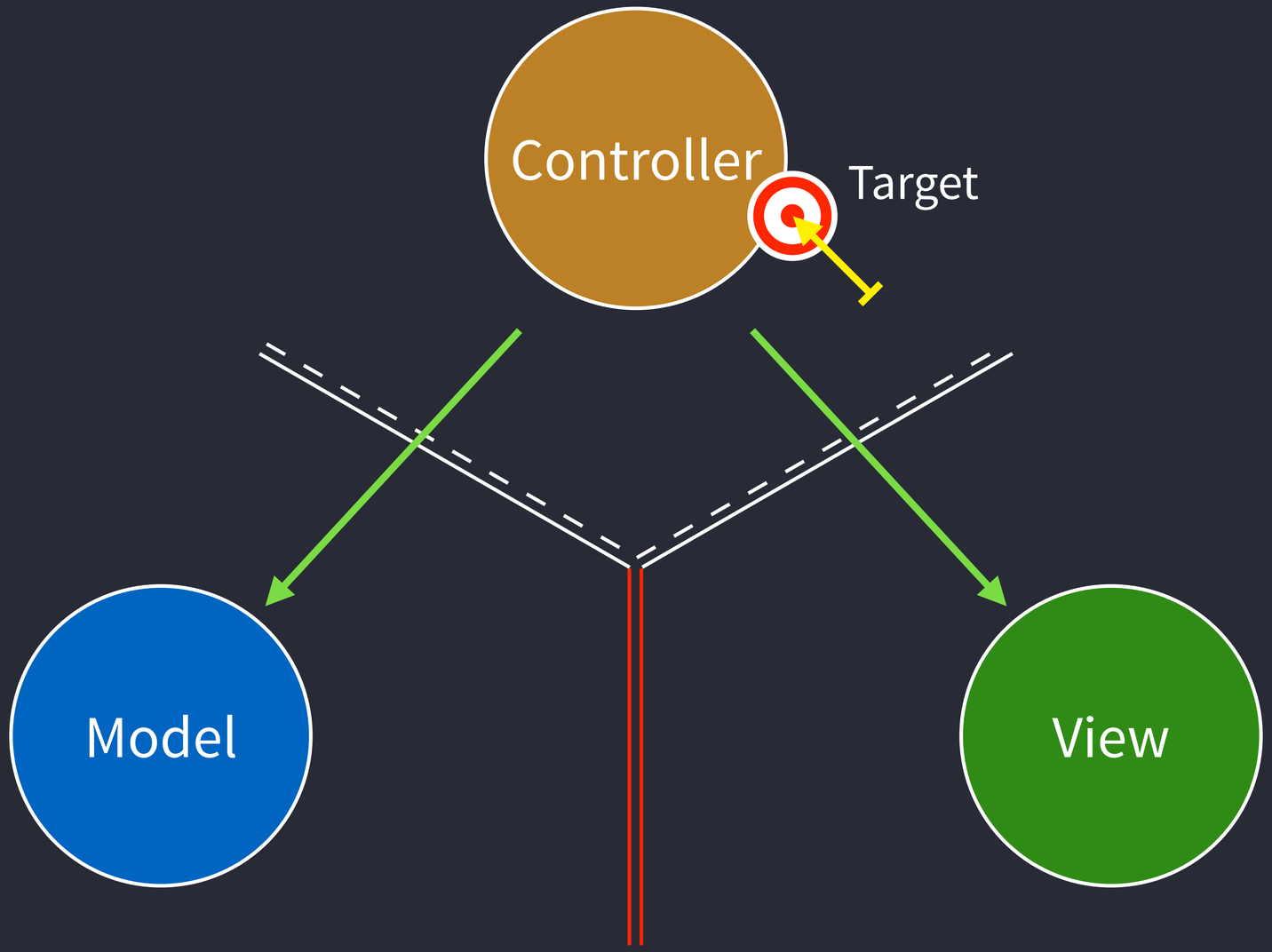
Model - View - Controller



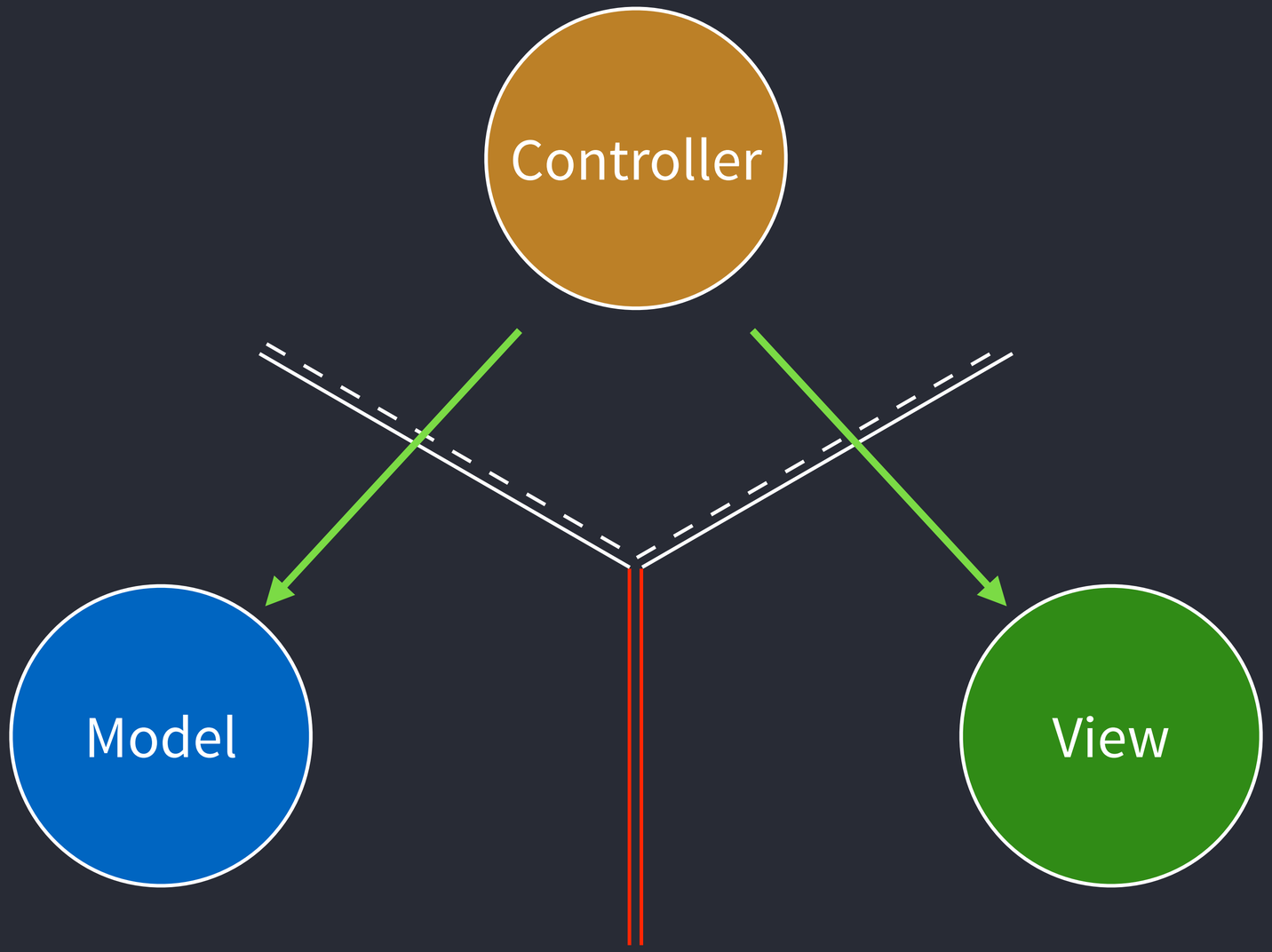
Model - View - Controller



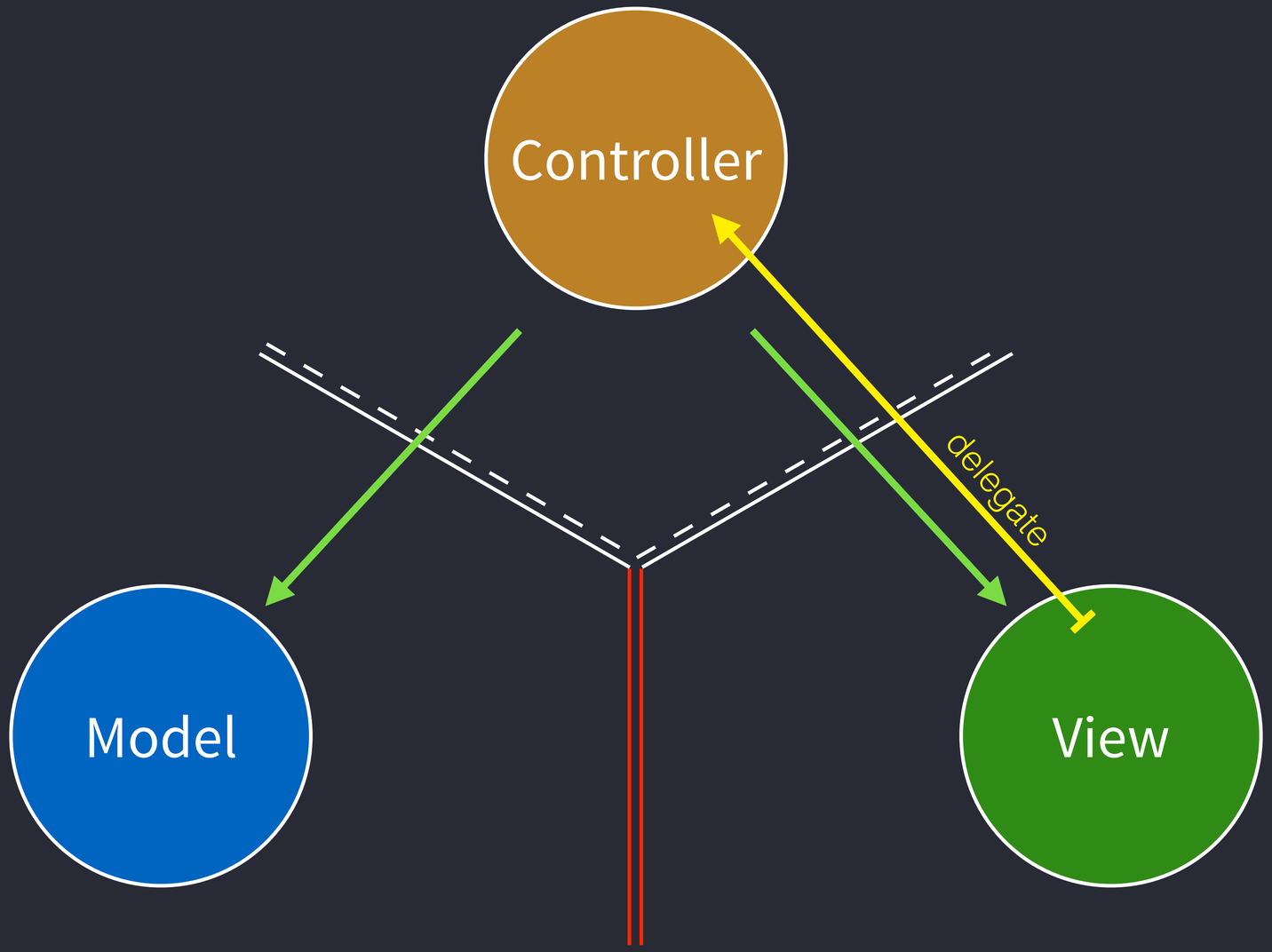
Model - View - Controller



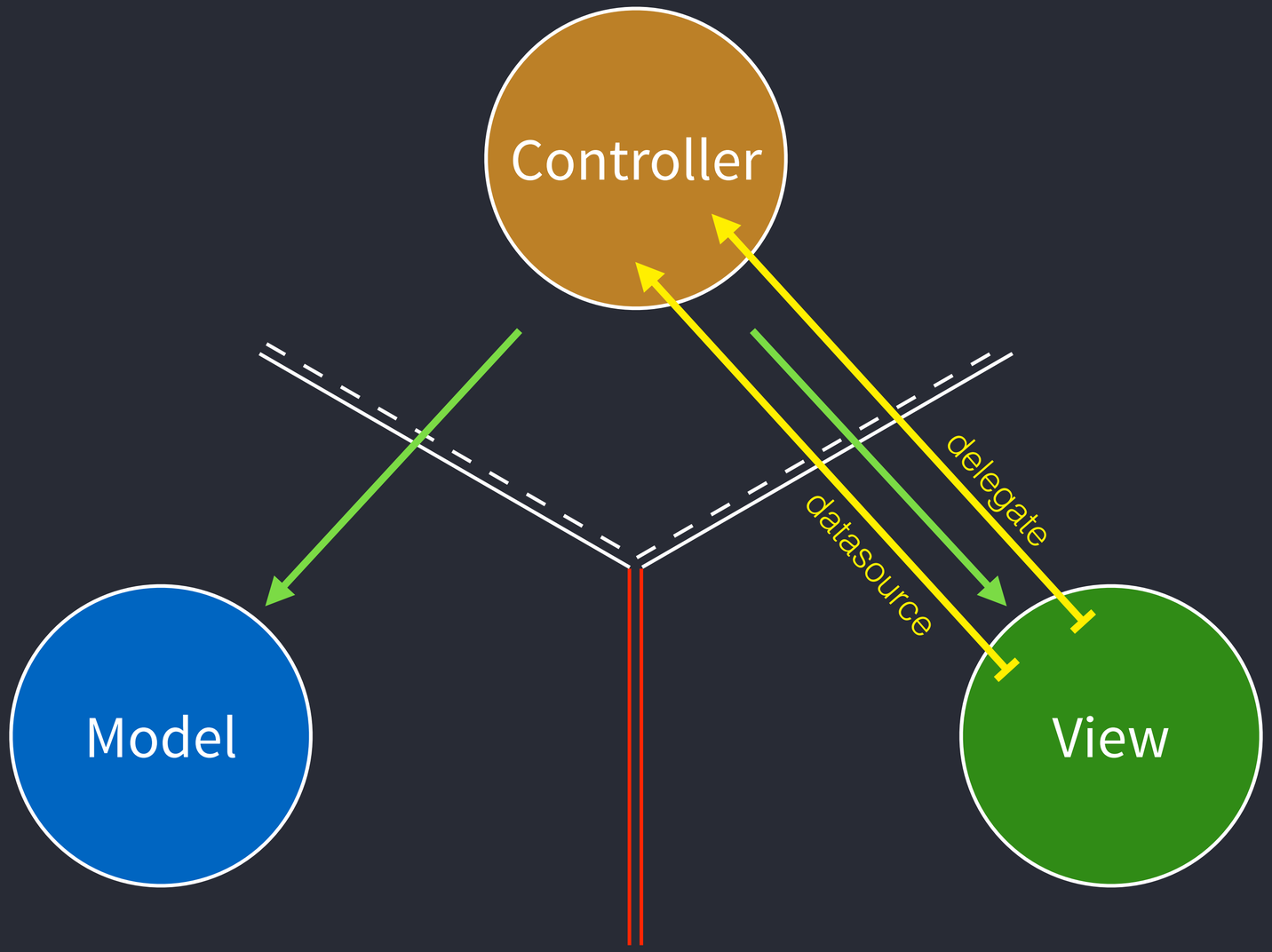
Model - View - Controller



Model - View - Controller



Model - View - Controller



Model - View - Controller

Model - View - Controller

Definition du delegate

```
@protocol MonSuperDelegate<NSObject>

- (void)didTouchCloseButton;
- (void)didSelectImage:(UIImage *)selectedImage;

@end


@interface MaVue : UIView

@property (weak) id<MonSuperDelegate> delegate;

@end
```

Model - View - Controller

Definition du delegate

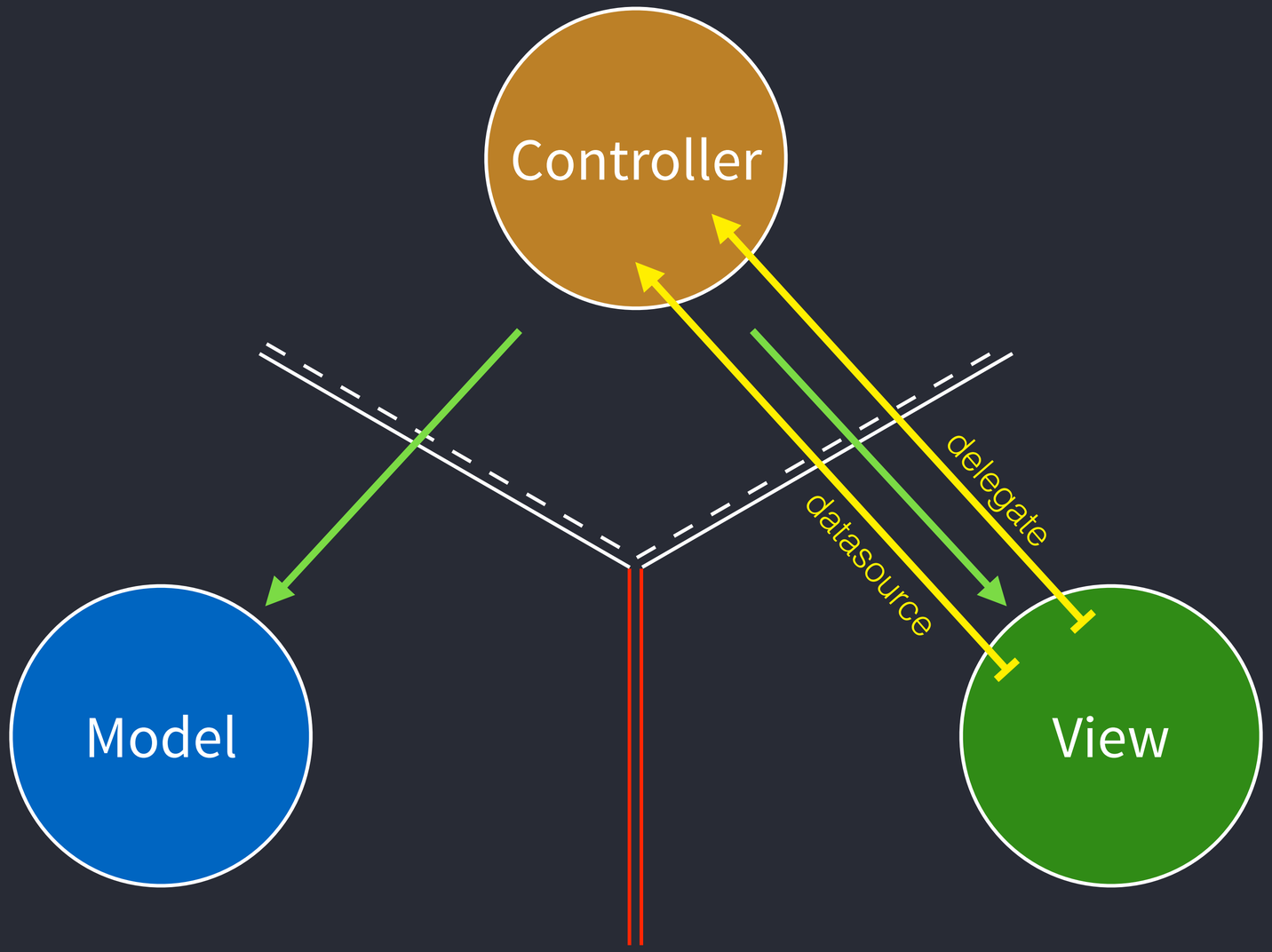
```
@protocol MonSuperDelegate<NSObject>

- (void)didTouchCloseButton;
- (void)didSelectImage:(UIImage *)selectedImage;

@end


@interface MaVue : UIView

@property (weak) id<MonSuperDelegate> delegate;

@end
```

Implementation du delegate

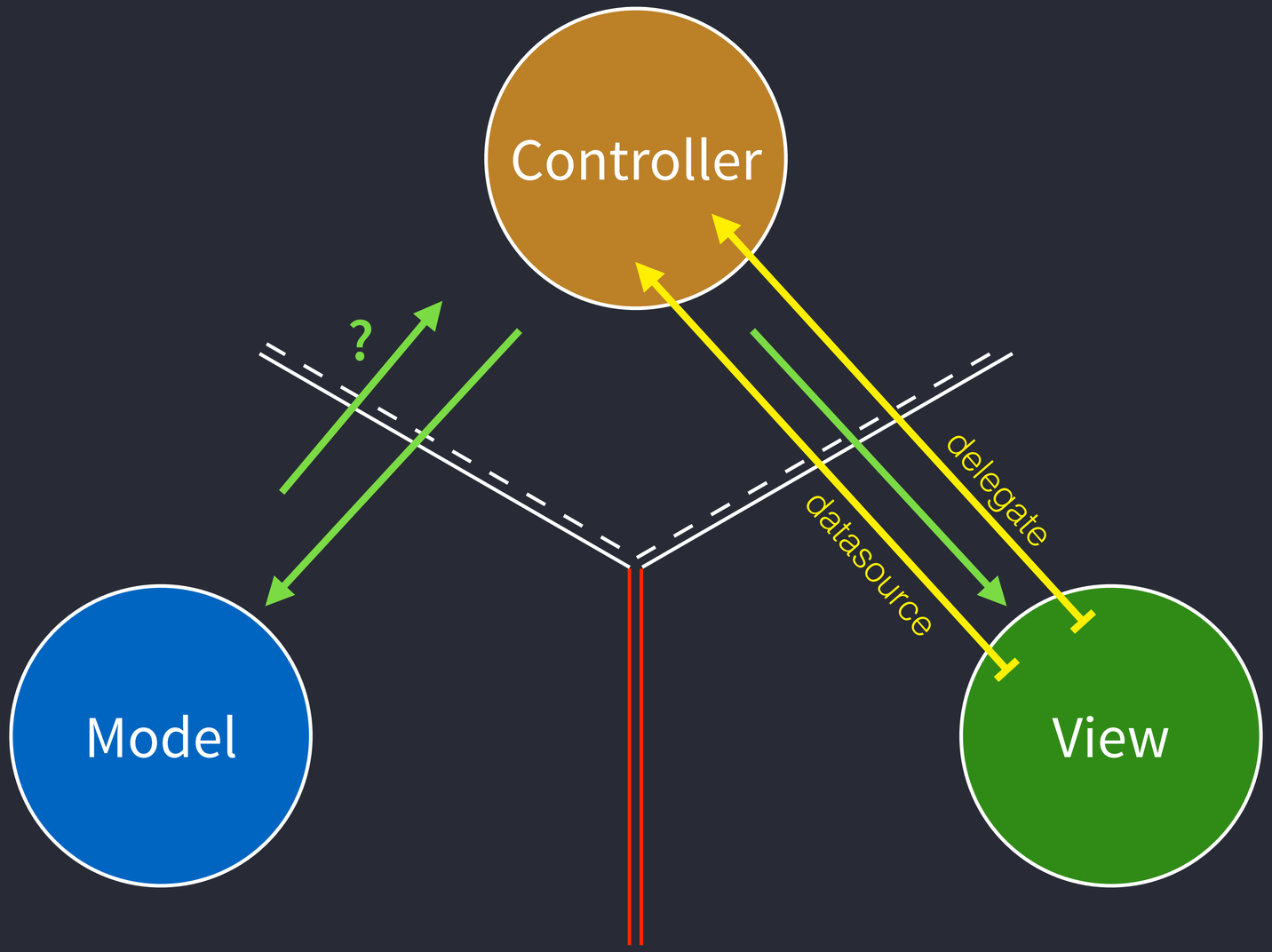
```
@interface MonController :
UIViewController<MonSuperDelegate>

@end
```

```
@implementation MonController

- (void)viewDidLoad {
   self.maVue.delegate = self;
}

#pragma mark - MonSuperDelegate

- (void)didTouchCloseButton {
    // ...
}

- (void)didSelectImage:(UIImage *)selectedImage {
    // ...
}

@end
```

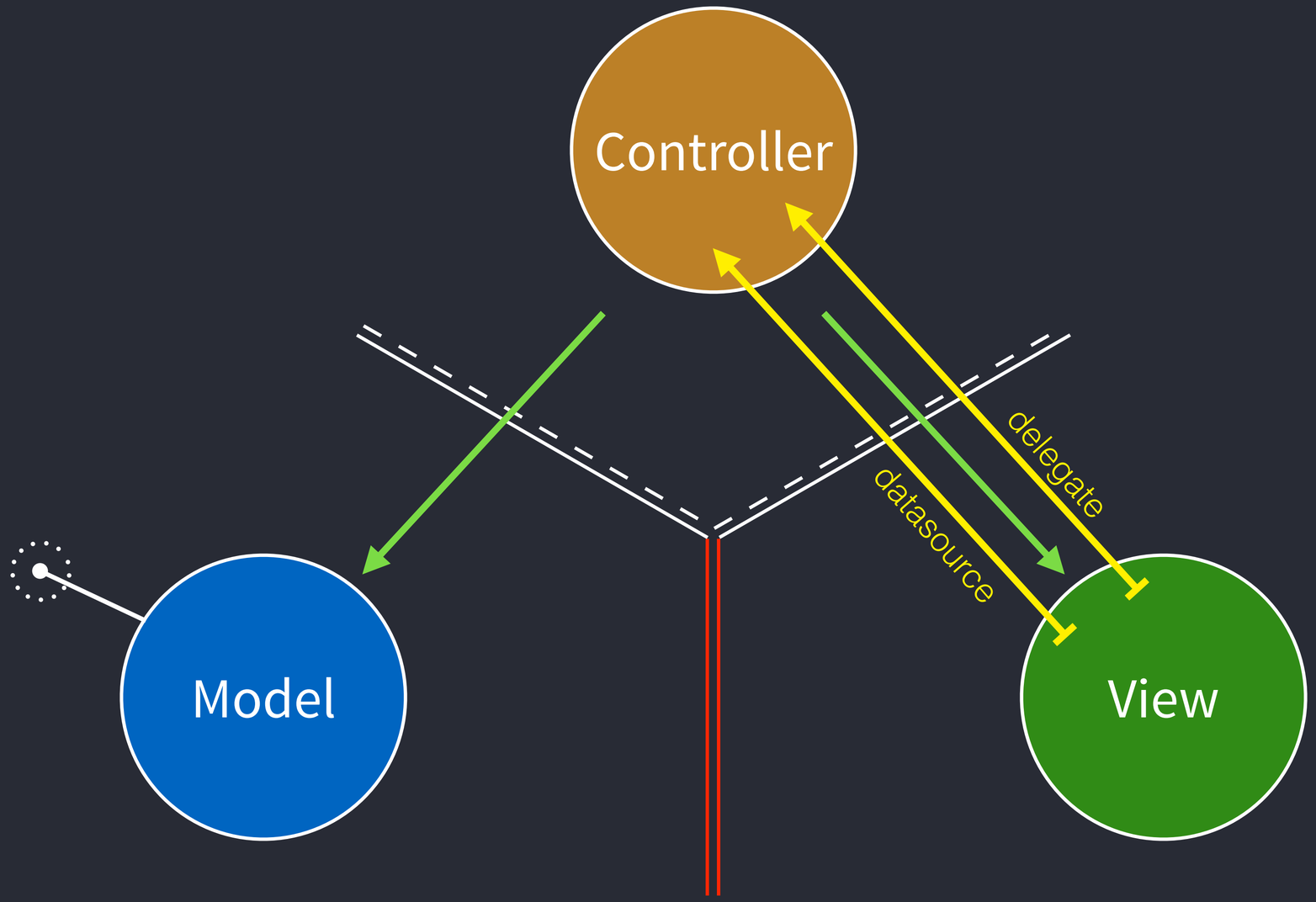
Model - View - Controller



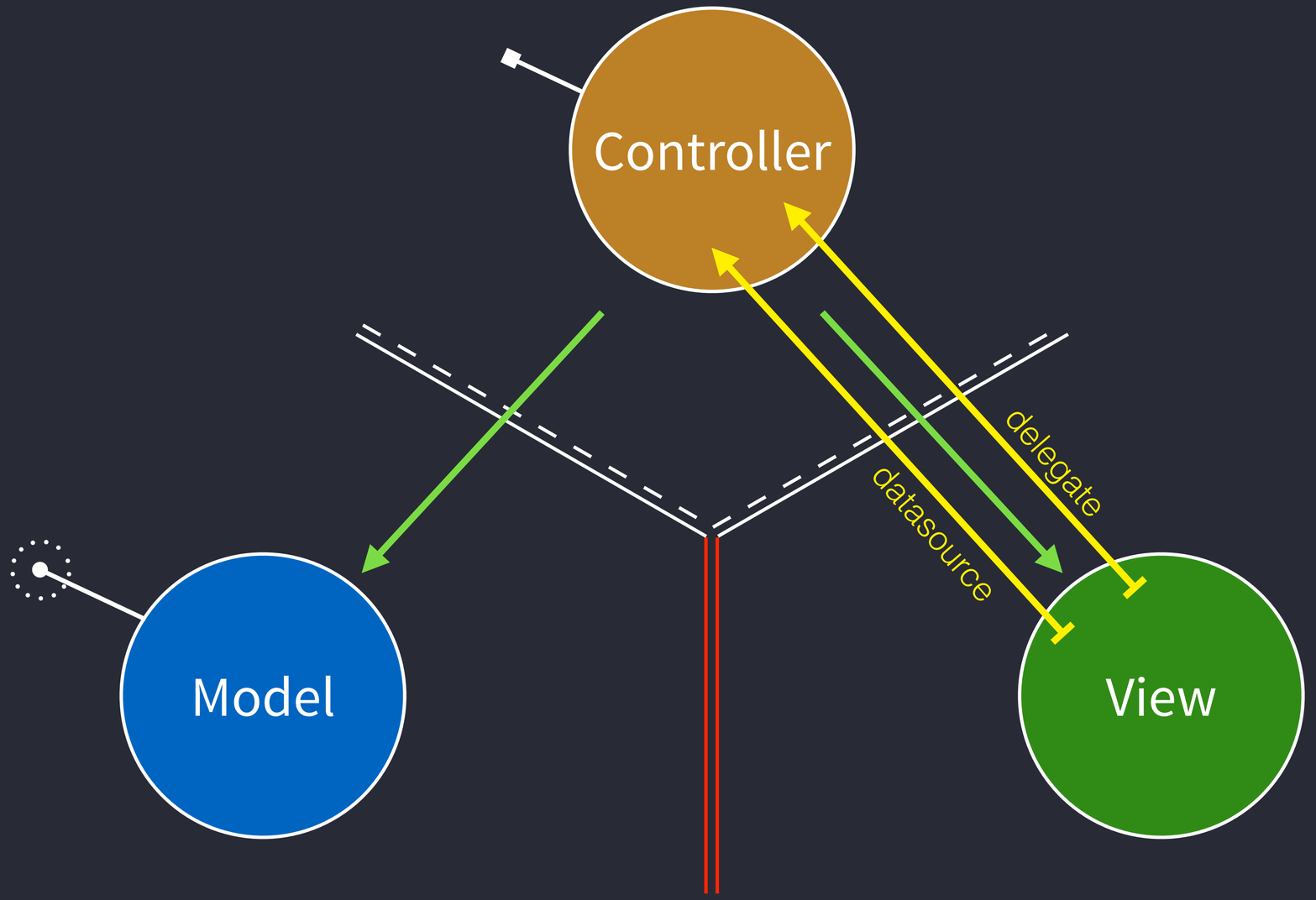
Model - View - Controller



Model - View - Controller



Model - View - Controller



Model - View - Controller

Implementation NSNotificationCenter

```
- (instancetype)init {
    self = [super init];
    if (self) {
        [[NSNotificationCenter defaultCenter] addObserver:self
                                                 selector:@selector(keyboardWillShow)
                                                     name:UIKeyboardWillShowNotification
                                                   object:nil];

    }
    return self;
}

- (void)keyboardWillShow {
    // Do stuffs...
}
```

Model - View - Controller

Implementation NSNotificationCenter

```
- (instancetype)init {
    self = [super init];
    if (self) {
        [[NSNotificationCenter defaultCenter] addObserver:self
                                                 selector:@selector(keyboardWillShow)
                                                     name:UIKeyboardWillShowNotification
                                                   object:nil];

    }
    return self;
}

- (void)keyboardWillShow {
    // Do stuffs...
}
```

```
- (void)dealloc {
   [[NSNotificationCenter defaultCenter] removeObserver:self
                                                   name:UIKeyboardWillShowNotification
                                                 object:nil];
}
```

Model - View - Controller

Implementation KVO (Key-Value Observing

```
void *KVOContext = &KVOContext;

- (void)viewDidLoad {
    [self.dataHolder addObserver:self
                      forKeyPath:@"totalValue"
                         options:NSKeyValueObservingOptionNew
                         context:KVOContext];
}

- (void)observeValueForKeyPath:(NSString *)keyPath ofObject:(id)object change:(NSDictionary *)change
context:(void *)context {
    if (context == KVOContext) {
        if ([keyPath isEqualToString:@"totalValue"]) {
            // Do stuffs…
        }
    }
}
```

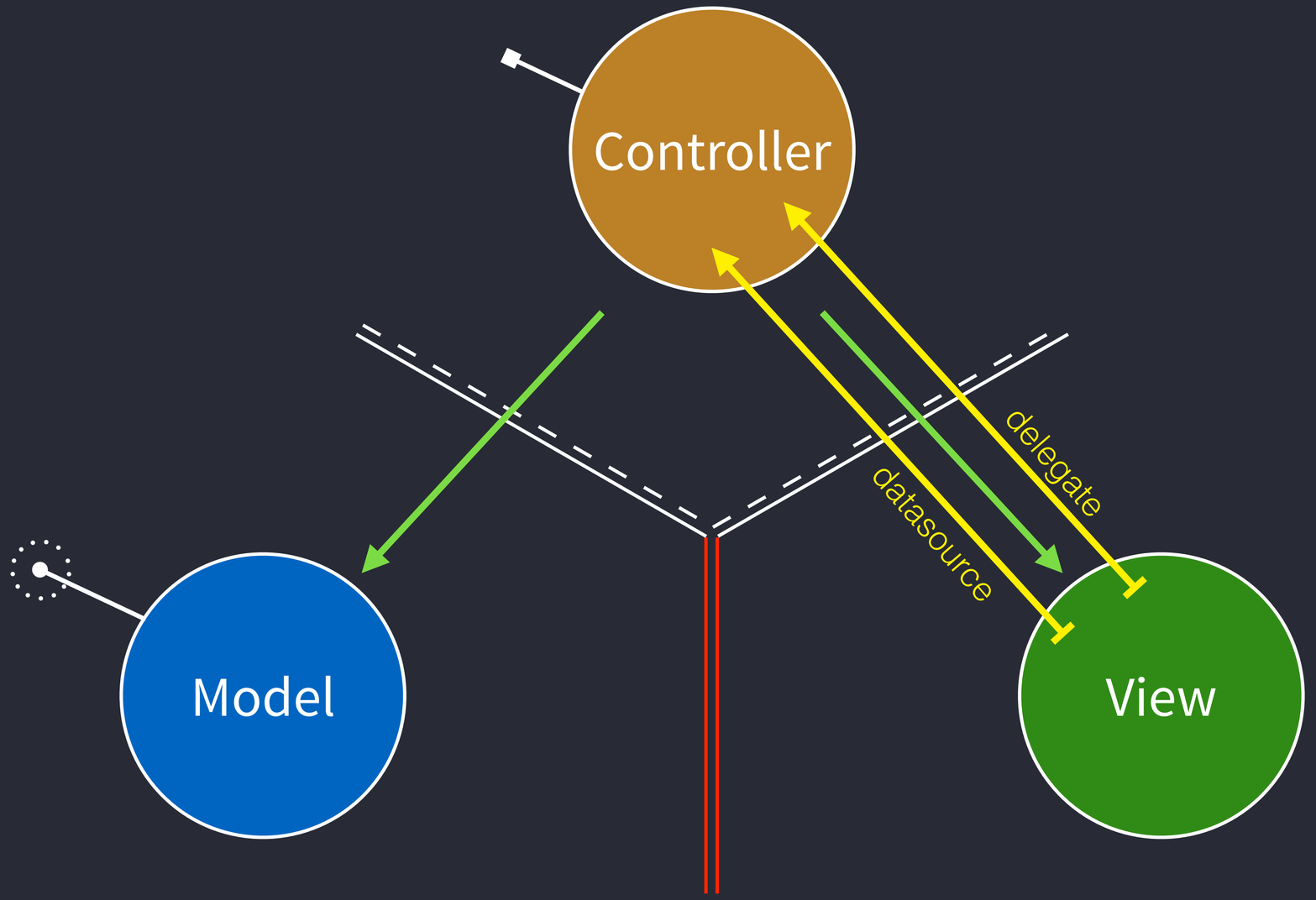
Model - View - Controller

Implementation KVO (Key-Value Observing

```
void *KVOContext = &KVOContext;

- (void)viewDidLoad {
    [self.dataHolder addObserver:self
                      forKeyPath:@"totalValue"
                         options:NSKeyValueObservingOptionNew
                         context:KVOContext];
}

- (void)observeValueForKeyPath:(NSString *)keyPath ofObject:(id)object change:(NSDictionary *)change
context:(void *)context {
    if (context == KVOContext) {
        if ([keyPath isEqualToString:@"totalValue"]) {
            // Do stuffs…
        }
    }
}
```

```
- (void)dealloc {
   [self.settingsHolder removeObserver:self
                            forKeyPath:@"totalValue"
                               context:KVOContext];
}
```

Model - View - Controller



Networking

- Les requêtes réseau sont la base de la plupart des applications mobiles.
- Il existe :
  - Des bibliothèques système (NSURLConnection…)
  - Des bibliothèques externes (AFNetworking…)

NSURLConnection

- NSURLConnection permet différents types de requêtes + de faire des reglages sur la requete :
  - synchrone/asynchrone
  - GET/POST
  - Timeout and cache policies
- Il existe d’autres API système, de plus bas niveau.

AFNetworking

- AFNetworking est une librairie Objective-C créée et maintenue par Matt Thompson. afnetworking.com
- Très simple à utiliser, c’est la librairie de networking la plus utilisée sur iOS.
- Syntaxe plus moderne que les API système.
- Permet de récupérer les données directement dans des formats exploitables (NSDictionary, NSArray, UIImage, etc.)

Démo