Payment in Moodle 3.10

Shamim Rezaie
Analyst Developer
The problem

How did it begin...
● Code duplication
● Code coupling

Only if we could have...
● Single plugin that support multiple payments
● Multiple plugin types can work with a centralised API without duplicating code

Parameters:
- <string> Merchant id
- <string> Order id (unique)
- <string> Extra information
- ...

Developing a payment consumer plugin

● **Trigger**

```html
<button
    data-action="core_payment/triggerPayment"
    data-component="availability_fee"
    data-paymentarea="area"
    data-itemid="1"
    data-cost="$180"
    data-description="Description goes here"
  >
    Send payment
  </button>
```

● **The service_provider class**

```php
namespace availability_fee\payment;

class service_provider implements \core_payment\local\callback\service_provider {
    public static function get_payable($paymentarea, $itemid): \core_payment\local\entities\payable {/* ... */}
    public static function deliver_order($paymentarea, $instanceid, $paymentid, $userid): bool {/* ... */}
    public static function get_success_url(string $paymentarea, int $instanceid): \moodle_url {/* ... */}
}
```
Developing a payment consumer plugin

- **Trigger**
  ```html
  <button
    data-action="core_payment/triggerPayment"
    data-component="availability_fee"
    data-paymentarea="area"
    data-itemid="1"
    data-cost="$180"
    data-description="Description goes here"
  >
    Send payment
  </button>
  ```

- **The service_provider class**
  ```php
  namespace availability_fee\payment;
  class service_provider implements \core_payment\local\callback\service_provider {
    public static function get_payable($paymentarea, $itemid): \core_payment\local\entities\payable {/* ... */}
    public static function deliver_order($paymentarea, $instanceid, $paymentid, $userid): bool {/* ... */}
    public static function get_success_url(string $paymentarea, int $instanceid): \moodle_url {/* ... */}
  }
  ```
Developing a new payment gateway (1/2)

payment/gateway/stripe

● The gateway class

```php
namespace paygw_stripe;

class gateway extends \core_payment\gateway {
    public static function get_supported_currencies(): array {/* ... */}
    public static function add_configuration_to_gateway_form($form): void {/* ... */}
    public static function validate_gateway_form($form,$data, array $files, array &$errors): void {/* ... */}
}
```

● The gateways_modal amd

```javascript
export const process = (component, paymentArea, itemId, description) => {
    // Returns a promise that is resolved if and when the payment and the order delivery are both successful.
}
```
Developing a new payment gateway (2/2)

- A web service or a return/notify php file

```php
$core_payment\helper::save_payment($accountid, $component, $paymentarea, $itemid, (int) $USER->id, $amount, $currency, stripe');

$core_payment\helper::deliver_order($component, $paymentarea, $itemid, (int) $USER->id);

// Only if in a return/notify php file.
redirect(
    $core_payment\helper::get_success_url($component, $paymentarea, $itemid),
    'Your transaction was successful.',
    null,
    $core\output\notification::NOTIFY_SUCCESS
);
```