Getting Started with the WePay APIs

Intro

WePay helps platforms increase revenue through integrated payments without friction or fraud. We provide everything you need to seamlessly facilitate payments between your users (payers and merchants, buyers and sellers, donors and organizers, etc.)

What to expect from this doc

This doc uses basic examples (easy to copy and paste) to demonstrate how the fundamentals of integrated payments work. Most examples utilize the minimum amount of required information and illustrate a straightforward path.

After reading this doc, you'll be able to:

- build a proof of concept (POC)

- start making basic API calls

Platform Setup

A “platform” is the digital space where your organization facilitates payments from payers to merchants. We’ll often refer to “your platform” in the docs. Your platform may be associated with multiple applications, which you can manage and configure in the Partner Center (Beta). Check out Platform Setup for more information.

Subscribe to payout notifications

Note: It’s very important to subscribe to all notifications during the initial setup of your application. They help you stay informed, monitor issues, and communicate with your users. In the code example below, we’re highlighting the importance of payout notifications.

Payouts are automatically disbursed to your merchant (based on their payout schedule and available funds). Because of this, it’s crucial your platform subscribes to payout notifications otherwise they may be triggered (or even fail) while your platform is unaware. Use the notification preferences resource to subscribe to payout notifications.

Argument

```
Request method: POST
Request path: /notification_preferences
Request headers:
   App-Id: 121212
   App-Token: prod_MTAwXzk5OWIwZT666LWYwNWItNDU4MS1iZjBiL
   Api-Version: 3.0
   Content-Type: application/json

Request body:

{
  "topic": "payouts.created",
  "callback_uri": "https://example.com/payouts"
}
```

Note: Subscribing to payouts.created will result in your platform receiving payout notifications when a new payout is created.

Response

Response


```
{
  "id": "6a838cf6-cddd-454d-ba3e-e0c4fbf6f9f2",
  "resource": "notification_preferences",
  "path": "/notification_preferences/6a838cf6-cddd-454d-ba3e-e0c4fbf6f9f2",
  "owner": {
      "id": "33e7673b-cd08-4579-bf8a-11af545d6e77",
      "resource": "applications",
      "path": null
  },
  "create_time": 1509407823974,
  "topic": "payouts.created",
  "callback_uri": "https://example.com/payouts",
  "state": "active"
}
```


More details on subscribing to notifications

Merchant Onboarding

Merchant onboarding is the process by which you enable your merchants (sellers, organizers, etc.) to accept payments. It involves collecting and verifying identity information that helps us protect your platform from fraud and risk. Legal entities help WePay collect information about a merchant so we can provision them an account. Accounts allow merchants to accept payments and pay out funds to their preferred payout method.

Merchant onboarding includes the following API resources:

- legal entities
- accounts
- payout methods

Legal Entities

Onboarding a merchant requires the creation of a legal entity. A legal entity is a person, business, or nonprofit organization. WePay collects information about the legal entity so we can properly underwrite them when provisioning an account.

Create a legal entity

Use the POST /legal_entities call to create a legal entity with just a country parameter.

Argument

```
Request method: POST
Request path: /legal_entities
Request headers:
   App-Id: 121212
   App-Token: prod_MTAwXzk5OWIwZT666LWYwNWItNDU4MS1iZjBiL
   Api-Version: 3.0
   Content-Type: application/json

Request body:

{
  "country": "US"
}
```

Response

```
{
    "terms_of_service": {
        "acceptance_time": null,
        "original_ip": null
    },
    "controller": {
        "name": null,
        "phone": null,
        "address": null,
        "email": null,
        "email_is_verified": false,
```

```
      "personal_country_info": {
         "US": {
            "social_security_number_last_four_is_present": false,
            "social_security_number_is_present": false
         }
      },
      "date_of_birth_is_present": false
   },
   "controller_is_beneficial_owner": null,
   "entity_name": null,
   "phone": null,
   "primary_url": null,
   "description": null,
   "address": null,
   "entity_country_info": {
      "US": {
         "legal_form": null,
         "employer_identification_number": null
      }
   },
   "additional_beneficial_owners": null,
   "custom_data": null,
   "country": "US",
   "create_time": 1525208689,
   "id": "f831964e-0850-4caa-967d-6d0040f82f4e",
   "resource": "legal_entities",
   "path": "/legal_entities/f831964e-0850-4caa-967d-6d0040f82f4e",
   "owner": {
      "id": ""33e7673b-cd08-4579-bf8a-11af545d6e77",
      "resource": "applications",
      "path": null
   }
}
```

More details on creating a legal entity

Accounts

Onboarding a merchant requires the creation of an account. Accounts have two fundamental capabilities: process a transaction and pay out funds received from a transaction. These

capabilities are enabled when WePay obtains enough information about the merchant via the legal entity and account.

Create a new account

Use the POST /accounts call to create an account using only the country parameter supplied in the previous example.

Note: An account is owned by a single legal entity.

Argument

```
Request method: POST
Request path: /accounts
Request headers:
   App-Id: 121212
   App-Token: prod_MTAwXzk5OWIwZT666LWYwNWItNDU4MS1iZjBiL
   Api-Version: 3.0
   Content-Type: application/json

Request body:

{
  "legal_entity_id": "f831964e-0850-4caa-967d-6d0040f82f4e"
}
```

Response

```
{
  "name": null,
  "description": null,
  "industry": {
    "merchant_category_code": null,
    "category_detail": null
  },
  "reference_id": null,
  "statement_description": null,
  "incoming_payments": {
    "accepted_methods": [
      "payment_bank",
```

```
      "visa",
      "mastercard",
      "american_express",
      "discover",
      "jcb",
      "diners_club"
    ]
  },
  "payout": {
    "default_currency": "USD",
    "currencies": {
      "USD": {
        "period": null,
        "payout_method_id": null
      }
    }
  },
  "custom_data": null,
  "create_time": 1516224518,
  "balances": {
    "currencies": {}
  },
  "id": "1ad00967-4812-4a98-aeb4-be38b2c704d3",
  "resource": "accounts",
  "path": "/accounts/1ad00967-4812-4a98-aeb4-be38b2c704d3",
  "owner": {
    "id": "f831964e-0850-4caa-967d-6d0040f82f4e",
    "resource": "legal_entities",
    "path": "/legal_entities/f831964e-0850-4caa-967d-6d0040f82f4e"
  }
}
```

More details on creating an account

Account Capabilities

You can look up an account's capabilities any time after creating it. WePay returns several pieces of information indicating whether payments or payouts are enabled and how to enable them if they're not. In most cases, you will need to supply account or legal entity information to enable a capability.

Look up account capabilities

Since your platform can create a legal entity and account with a small amount of merchant information, use the GET /accounts/{id}/capabilities call to ask WePay exactly what information the merchant needs to supply in order to enable payments and payouts.

Argument

```
Request method: GET
Request path:
/accounts/1ad00967-4812-4a98-aeb4-be38b2c704d3/capabilities?is_expanded=true
Request headers:
  App-Id: 121212
  App-Token:
prod_MTAwXzk5OWIwZTA2LWYwNWItNDU4MS1iZjBiLTU2N2MxMDExOTIyMQ
  Api-Version: 3.0
  Content-Type: application/json
```

Response


```
{
   "payments": {
      "enabled": false,
      "current_issues": [
         {
            "errant_fields": {
               "name": [
                  "is_null"
               ]
            },
            "issue_type": "errant_fields",
            "target": {
               "resource": "accounts",
               "id": "1ad00967-4812-4a98-aeb4-be38b2c704d3",
               "path": "/accounts/1ad00967-4812-4a98-aeb4-be38b2c704d3"
            }
         }
      ],
      "upcoming_issues": {}
```

```
  },
  "payouts": {
    "enabled": false,
    "current_issues": [
      {
        "errant_fields": {
          "name": [
            "is_null"
          ],
          "payouts.currencies.USD.payout_method_id": [
            "is_null"
          ],
          "payouts.currencies.USD.period": [
            "is_null"
          ]
        },
        "issue_type": "errant_fields",
        "target": {
          "resource": "accounts",
          "id": "1ad00967-4812-4a98-aeb4-be38b2c704d3",
          "path": "/accounts/1ad00967-4812-4a98-aeb4-be38b2c704d3"
        }
      },
      {
        "errant_fields": {
          "controller.phone": [
            "is_null"
          ]
        },
        "issue_type": "errant_fields",
        "target": {
          "resource": "legal_entities",
          "id": "f831964e-0850-4caa-967d-6d0040f82f4e",
          "path": "/legal_entities/f831964e-0850-4caa-967d-6d0040f82f4e"
        }
      }
    ],
    "upcoming_issues": {}
  },
  "id": null,
  "resource": "capabilities",
  "path": "/accounts/1ad00967-4812-4a98-aeb4-be38b2c704d32/capabilities",
```

```
    "owner": {
        "id": "1ad00967-4812-4a98-aeb4-be38b2c704d3",
        "resource": "accounts",
        "path": "/accounts/1ad00967-4812-4a98-aeb4-be38b2c704d3"
    }
}
```

More details on looking up an account's capabilities

Update a legal entity

The flexibility of the WePay API allows your platform to update a legal entity any time after creating it using the POST /legal_entities/{id}.

Note: Updating a legal entity's information may impact an account's capabilities.

Argument

```
Request method: POST
Request path: /legal_entities/f831964e-0850-4caa-967d-6d0040f82f4e
Request headers:
   App-Id: 121212
   App-Token: prod_MTAwXzk5OWIwZT666LWYwNWItNDU4MS1iZjBiL
   Api-Version: 3.0
   Content-Type: application/json

Request body:

{
  "controller": {
      "phone": {
        "country_code": "+1",
        "phone_number": "5555555555"
      }
  }
}
```

```
{
  "resource": "legal_entities",
  "id": "f831964e-0850-4caa-967d-6d0040f82f4e",
  "path": "/legal_entities/f831964e-0850-4caa-967d-6d0040f82f4e",
  "owner": {
    "resource": "applications",
    "id": ""33e7673b-cd08-4579-bf8a-11af545d6e77",
    "path": null
  },
  "create_time": 1480700534,
  "country": "US",
  "terms_of_service": {
    "acceptance_time": null,
    "original_ip": null
  },
  "entity_name": null,
  "controller_is_beneficial_owner": null,
  "controller": {
    "name": null,
    "phone": {
       "country_code": "+1",
       "phone_number": "5555555555"
    },
    "address": null,
    "email": null,
    "job_title": null,
    "personal_country_info": {}
  },
  "entity_name": null,
  "phone": null,
  "primary_url": null,
  "description": null,
  "address": null,
  "entity_country_info": {},
  "additional_beneficial_owners": null,
  "custom_data": null
}
```


More details on updating a legal entity

Payout Methods

Onboarding a merchant requires the creation of a payout method so merchants can receive the payments they've accepted. Payout methods indicate the way account funds are disbursed (i.e bank account) to merchants and are used by a legal entity's account to perform payouts.

Create a payout method

Use the POST /payout_methods call to create a payout method.

Note: Currently, the only payout method is a bank account and a payout method is owned by a single legal entity.

Argument

```
Request method: POST
Request path: /payout_methods
Request headers:
   App-Id: 121212
   App-Token: prod_MTAwXzk5OWIwZT666LWYwNWItNDU4MS1iZjBiL
   Api-Version: 3.0
   Content-Type: application/json

Request body:

{
  "legal_entity_id": "f831964e-0850-4caa-967d-6d0040f82f4e",
  "nickname": "My BofA account",
  "type": "payout_bank_us",
  "payout_bank_us": {
    "account_number": "123456789",
    "routing_number": "021000021",
    "account_type": "checking"
  }
}
```

Response


```
{
  "resource": "payout_methods",
  "id": "d64f3abd-1146-4281-af79-ac094419beff",
  "path": "/payout_methods/d64f3abd-1146-4281-af79-ac094419beff",
  "owner": {
    "resource": "legal_entities",
    "id": "f831964e-0850-4caa-967d-6d0040f82f4e",
    "path": "/legal_entities/f831964e-0850-4caa-967d-6d0040f82f4e"
  },
  "create_time": 1480700534,
  "nickname": "My BofA account",
  "type": "payout_bank_us",
  "payout_bank_us": {
    "account_number": "123456789",
    "routing_number": "021000021",
    "account_type": "checking"
  },
  "custom_data": null
}
```


More details on creating a payout method

Update an account

The flexibility of the WePay API allows your platform to update an account any time after creating it. Once you create a payout method for the merchant, you can add the payout method ID and payout period (schedule) to the account. In addition, you can supply the missing information discovered while querying the account's capabilities.

Use the POST /accounts/{id} call to update an account.

Argument

```
Request method: POST
Request path: /accounts/1ad00967-4812-4a98-aeb4-be38b2c704d3
Request headers:
```

```
  App-Id: 121212
  App-Token: prod_MTAwXzk5OWIwZT666LWYwNWItNDU4MS1iZjBiL
  Api-Version: 3.0
  Content-Type: application/json

Request body:

{
  "name": "Scout’s account",
  "payout": {
    "currencies": {
      "USD": {
        "payout_method_id": "d64f3abd-1146-4281-af79-ac094419beff",
        "period": "daily"
      }
    }
  }
}
```

Response

```
{
  "resource": "accounts",
  "id": "1ad00967-4812-4a98-aeb4-be38b2c704d3",
  "path": "/accounts/1ad00967-4812-4a98-aeb4-be38b2c704d3",
  "owner": {
    "resource": "legal_entities",
    "id": "f831964e-0850-4caa-967d-6d0040f82f4e",
    "path": "/legal_entities/f831964e-0850-4caa-967d-6d0040f82f4e"
  },
  "create_time": 1480700534,
  "incoming_payments": {
    "enabled": true,
    "accepted_methods": [
      "payment_bank",
      "visa",
      "mastercard",
      "american_express",
      "discover",
```

```
      "jcb",
      "diners_club"
    ]
  },
  "balances": {},
  "payout": {
    "default_currency": "USD",
    "currencies": {
      "USD": {
        "payout_method_id": "d64f3abd-1146-4281-af79-ac094419beff",
        "period": "daily",
        "next_payout_time": 14987382982
      }
    }
  }
}
```


More details on updating an account

Look up an account’s capabilities

Once your platform supplies missing information, the GET /accounts/{id}/capabilities call will indicate that payments and payouts are enabled.


```
{
    "payments": {
        "enabled": true,
        "current_issues": [],
        "upcoming_issues": {}
    },
    "payouts": {
        "enabled": true,
        "current_issues": [],
        "upcoming_issues": {}
    },
    "id": null,
    "resource": "capabilities",
    "path": "/accounts/1ad00967-4812-4a98-aeb4-be38b2c704d3/capabilities",
    "owner": {
        "id": "1ad00967-4812-4a98-aeb4-be38b2c704d3",
        "resource": "accounts",
```

```
      "path": "/accounts/1ad00967-4812-4a98-aeb4-be38b2c704d3"
    }
}
```

More details on looking up an account's capabilities

Payments

Once you onboard a merchant, they can begin accepting funds. Payments facilitate the movement of money from the payer to the merchant. Tokenization helps protect sensitive payment data when funds move from the payer to the merchant. If necessary, a refund may be created by providing the ID of the associated payment instance.

Tokens

A token is a unique placeholder, which you can generate via our JavaScript library, that replaces real-life credit card information in your system.

Tokenize payment data

WePay provides a JavaScript library to allow for easier development of short-lived tokens. Use the Tokenization JavaScript library [link – coming soon] to tokenize payment information such as credit cards, bank accounts, and digital wallet payments.

```
WePay.tokens.create({
  "type": "payment_methods",
  "payment_methods": {
    "type": "credit_card",
    "credit_card": {
      "card_holder": {
        "holder_name": "Jon Snow",
        "email": "jsnow@stark.com",
        "address": {
          "country": "US",
          "postal_code": "94025"
        }
      },
      "card_number": "5496198584584769",
      "expiration_month": 4,
      "expiration_year": 2023,
      "cvv": "007",
      "virtual_terminal_mode": "web",
      "auto_update": false
    }
  }
}, {}, function(response) {
  console.log(response);
```

});

```
{
  "id": "5eb21dff-b917-4bdc-924c-fec4c1b5790f",
  "resource": "tokens",
  "path": null,
  "owner": {
    "id": "33e7673b-cd08-4579-bf8a-11af545d6e77",
    "resource": "applications",
    "path": null
  },
  "create_time": 1512680838,
  "expire_time": 1512682638
}
```

More details on tokens

Payments

Accepting funds requires the creation of a payment. Payments facilitate the movement of money from the payer to the merchant and represent the instance of a payment. The payment is the most important part of accepting funds because it moves money from a payer to a merchant.

Create a new payment

Use the POST /payments call to create a payment.

Argument

Request method: POST
Request path: /payments
Request headers:
App-Id: 121212
App-Token: prod_MTAwXzk5OWIwZT666LWYwNWItNDU4MS1iZjBiL
Api-Version: 3.0
Content-Type: application/json

Request body:


```
{
  "account_id": "1ad00967-4812-4a98-aeb4-be38b2c704d3"
  "amount": 1000,
  "currency": "USD",
  "payment_method": {
    "token”: {
       "id": "5eb21dff-b917-4bdc-924c-fec4c1b5790f"
    }
```

```
  }
}
```

Response


```
{
  "id": "a7e5f113-8809-43e2-9617-dd8284767bc6",
  "resource": "payments",
  "path": "/payments/a7e5f113-8809-43e2-9617-dd8284767bc6",
  "owner": {
    "id": "1ad00967-4812-4a98-aeb4-be38b2c704d3",
    "resource": "accounts",
    "path": "/accounts/1ad00967-4812-4a98-aeb4-be38b2c704d3"
  },
  "create_time": 1510079535,
  "status": "completed",
  "amount": 1000,
  "currency": "USD",
  "payment_method": {
    "id": "d6f37226-c859-4efb-b1a8-5589a8d12126",
    "resource": "payment_methods",
    "path": "/payment_methods/d6f37226-c859-4efb-b1a8-5589a8d12126"
  },
  "failure_reason": null,
  "txnr_app_fee": null,
  "txnr_merchant": null,
  "signature_uri": "example.com",
  "amount_refundable": 1000
}
```

More details on creating a payment

Refunds
Refunds facilitate the return of money from the merchant to the payer. Refunds require the payment ID of a specific payment that’s been accepted.

Facilitate a refund
Use the POST /refunds call to create a refund.

Argument

Request method: POST
Request path: /refunds
Request headers:
  App-Id: 121212

App-Token: prod_MTAwXzk5OWIwZT666LWYwNWItNDU4MS1iZjBiL
Api-Version: 3.0
Content-Type: application/json

Request body:


```
{
  "payment_id": "a7e5f113-8809-43e2-9617-dd8284767bc6",
  "refund_reason": "Item(s) returned.",
  "order_id": "bedfbcbb-c0f0-41ea-aa33-61405f1c04eb"
}
```


Response


```
{
  "id": "f786a714-401b-4296-b02e-3f7203f06d75",
  "resource": "refunds",
  "path": "/refunds/f786a714-401b-4296-b02e-3f7203f06d75",
  "owner": {
    "id": "1ad00967-4812-4a98-aeb4-be38b2c704d3",
    "resource": "accounts",
    "path": "/accounts/1ad00967-4812-4a98-aeb4-be38b2c704d3"
  },
  "create_time": 1509140695,
  "status": "completed",
  "payment": {
    "id": "a7e5f113-8809-43e2-9617-dd8284767bc6",
    "resource": "payments",
    "path": "/payments/a7e5f113-8809-43e2-9617-dd8284767bc6"
  },
  "payment_method": {
    "id": "6ca5694b-4f5d-4d27-ad6d-5be93953bdcd",
    "resource": "payment_methods",
    "path": "/payment_methods/6ca5694b-4f5d-4d27-ad6d-5be93953bdcd"
  },
  "refund_reason": "Item(s) returned.",
  "amounts": {
    "total_amount": 100,
    "currency": "USD",
    "fee_refund_amount": 0
  },
  "pending_reasons": null,
  "failure_reason": null,
  "txnr_merchant_refund": {
```

```
    "id": "8172b463-82eb-4eae-9611-865a223ddf37",
    "resource": "transaction_records",
    "path": "/transaction_records/8172b463-82eb-4eae-9611-865a223ddf37"
  },
  "txnr_app_fee_refund": null,
  "order": null,
  "custom_data": null
}
```

More details on creating a refund

Payouts

Payouts allow merchants to receive their funds. Once a payment goes through, WePay will trigger a payout to disburse funds (based on the merchant's payout schedule and available funds) to the payout method set up by the merchant. Because of this, it's very important your platform subscribes to payout notifications during your application setup.

For more details, visit Payouts.

What's next

Check out Platform Setup for details on registering and configuring your applications.

Ready for the API? Head over to the API Reference.

Need help?

We're here for you. Please reach out to our developer support team at api@wepay.com with any questions or concerns.