Automatically Generate a PDF and send it by Email

File name: example.py

```
=============================================
#!/usr/bin/env python3
import emails
import os
import reports
table_data=[
  ['Name', 'Amount', 'Value'],
  ['elderberries', 10, 0.45],
  ['figs', 5, 3],
  ['apples', 4, 2.75],
  ['durians', 1, 25],
  ['bananas', 5, 1.99],
  ['cherries', 23, 5.80],
  ['grapes', 13, 2.48],
  ['kiwi', 4, 0.49]]
reports.generate("/tmp/report.pdf", "A Complete Inventory of My Fruit", "This is all my fruit.",
table_data)
sender = "automation@example.com"
receiver = "{}@example.com".format(os.environ.get('USER'))
subject = "List of Fruits"
body = "Hi\n\nI'm sending an attachment with all my fruit."
message = emails.generate(sender, receiver, subject, body, "/tmp/report.pdf")
emails.send(message)
```

File name: cars.py

```
=============================================
#!/usr/bin/env python3

import collections
import json
import locale
import mimetypes
import os.path
import reports
import emails
import sys
```

```
def load_data(filename):
  """Loads the contents of filename as a JSON file."""
  with open(filename) as json_file:
    data = json.load(json_file)
  return data


def format_car(car):
  """Given a car dictionary, returns a nicely formatted name."""
  return "{} {} ({})".format(
      car["car_make"], car["car_model"], car["car_year"])


def process_data(data):
  """Analyzes the data, looking for maximums.

  Returns a list of lines that summarize the information.
  """
  locale.setlocale(locale.LC_ALL, 'en_US.UTF8')
  max_sales = {"total_sales": 0}
  max_revenue = {"revenue": 0}
  car_year_sales = collections.defaultdict(int)
  for item in data:
    # We need to convert "$1234.56" into 1234.56
    item_price = locale.atof(item["price"].strip("$"))
    item_revenue = item["total_sales"] * item_price
    if item_revenue > max_revenue["revenue"]:
      item["revenue"] = item_revenue
      max_revenue = item

    if item["total_sales"] > max_sales["total_sales"]:
      max_sales = item
    car_year_sales[item["car"]["car_year"]] += item["total_sales"]

  max_car_sales_year = (0,0)
  for year, sales in car_year_sales.items():
    if sales > max_car_sales_year[1]:
      max_car_sales_year = (year,sales)
  summary = []
  summary.append("The {} generated the most revenue: ${}".format(
      format_car(max_revenue["car"]), max_revenue["revenue"]))
  summary.append("The {} had the most sales: {}".format(
      format_car(max_sales["car"]), max_sales["total_sales"]))
  summary.append("The most popular year was {} with {} sales.".format(
      max_car_sales_year[0], max_car_sales_year[1]))
  return summary


def cars_dict_to_table(car_data):
  """Turns the data in car_data into a list of lists."""
  table_data = [["ID", "Car", "Price", "Total Sales"]]
  for item in car_data:
    table_data.append([item["id"], format_car(item["car"]), item["price"], item["total_sales"]])
  return table_data


def main(argv):
  data = load_data(os.path.expanduser('~') + "/car_sales.json")
  summary = process_data(data)

  # Generate a paragraph that contains the necessary summary
  paragraph = "<br/>".join(summary)
  # Generate a table that contains the list of cars
  table_data = cars_dict_to_table(data)
  # Generate the PDF report
  title = "Sales summary for last month"
  attachment = "/tmp/cars.pdf"
  reports.generate(attachment, title, paragraph, table_data)

  # Send the email
  sender = "automation@example.com"
  receiver = "{}@example.com".format(os.environ.get('USER'))
  body = "\n".join(summary)
  message = emails.generate(sender, receiver, title, body, attachment)
  emails.send(message)


if __name__ == "__main__":
  main(sys.argv)
```